JP9034723

Publication Title:

COMMITMENT OF INTRODUCTION PLAN OBJECT FOR INTRODUCTION OF APPLICATION PROGRAM AT NETWORK

Abstract:

Abstract of JP 9034723

(A) Translate this text PROBLEM TO BE SOLVED: To introduce an application on a network by inspecting the suitability of an introduction schedule object in advance and transforming that object into data structure usable for a network introduction engine. SOLUTION: A commit process is started by selecting any commit menu out of an introduction schedule menu or the like and in a step 405, the introduction schedule object prepares an error log object for storing the respective messages of conditions, warning and error detected during the commit process. Next, in a step 410, when it is necessary to properly perform introduction or constitution, any object or attribute is added to the schedule object. Next, in a step 415, it is inspected whether proper data exist in the schedule or not. At a judge block 420, an error counter is inspected and when counting is '0', processing is moved to the part after a step 425 so that the object can be transformed into the data structure usable for the network introduction engine of application introduction.

Courtesy of http://v3.espacenet.com

This Patent PDF Generated by Patent Fetcher(R), a service of Stroke of Color, Inc.

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（Ａ）**　(11)特許出願公開番号

特開平9－34723

(43)公開日　平成９年(1997)２月７日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｇ０６Ｆ　9/445			Ｇ０６Ｆ　9/06	４２０Ｊ	
9/06	４１０			４１０Ｂ	

審査請求　未請求　請求項の数25　ＯＬ　（全 46 頁）

(21)出願番号　特願平8－82910

(22)出願日　平成８年(1996)４月４日

(31)優先権主張番号　４１７１６２

(32)優先日　1995年４月５日

(33)優先権主張国　米国（ＵＳ）

(71)出願人　390009531
インターナショナル・ビジネス・マシーンズ・コーポレイション
ＩＮＴＥＲＮＡＴＩＯＮＡＬ　ＢＵＳＩＮＥＳＳ　ＭＡＳＣＨＩＮＥＳ　ＣＯＲＰＯＲＡＴＩＯＮ
アメリカ合衆国10504、ニューヨーク州　アーモンク　（番地なし）

(72)発明者　ジョン・ローレンス・バンス
アメリカ合衆国78590　テキサス州オースチン　バトンウッド　11500

(74)代理人　弁理士　合田　潔　（外２名）

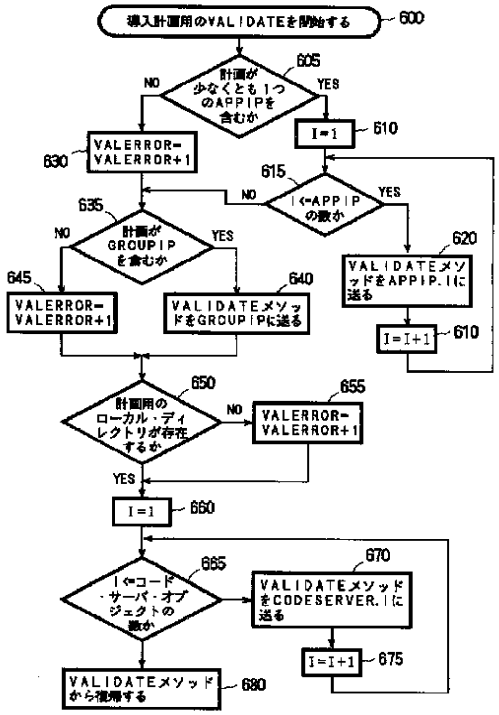
最終頁に続く

(54)【発明の名称】　ネットワークでのアプリケーション・プログラムの導入のための導入計画オブジェクトのコミット

(57)【要約】　（修正有）

【課題】　導入計画オブジェクトを妥当性検査する。

【解決手段】　導入計画オブジェクトは、プログラムを表すアプリケーション・オブジェクトと、そのアプリケーション・プログラムを導入するワークステーションのグループを表すオブジェクトとを含む。コミット・プロセスの一部として、導入計画オブジェクトは、その子オブジェクトを検査し、必要であれば導入計画オブジェクトに子オブジェクトを追加することによって事前妥当性検査され、導入計画オブジェクトとその子オブジェクトのデータにエラーがないかどうか検査することによって妥当性検査され、ネットワーク内でアプリケーションを導入するネットワーク導入エンジンに使用可能なデータ構造に変換される。そのアプリケーションの導入が応答ファイルと、そのデータをカストマイズするためのカストマイズ・ファイル・オブジェクトとを必要とする場合、導入計画は応答ファイル・オブジェクトをさらに含む。



【特許請求の範囲】
【請求項１】ネットワークでアプリケーションを導入するための導入計画オブジェクトをコミットする方法において、
導入計画オブジェクト内の子オブジェクトを検査し、必要であれば、導入計画オブジェクトに追加の子オブジェクトを追加することにより、導入計画オブジェクトを事前妥当性検査するステップと、
導入計画オブジェクトとその子オブジェクト内のデータを検査して、データにエラーがないかどうか確認することにより、導入計画オブジェクトを妥当性検査するステップと、
導入計画が正常に妥当性検査された場合に、導入計画オブジェクト内の子オブジェクトを、データ構造に応じてネットワークでアプリケーションを導入するネットワーク導入エンジンに使用可能なデータ構造に変換するステップとを含む方法。
【請求項２】導入計画オブジェクトが、アプリケーション・プログラムを表す計画中アプリケーション・オブジェクトと、アプリケーション・プログラムを導入すべきワークステーションのグループを表す計画中グループ・オブジェクトとを含むことを特徴とする、請求項１に記載の方法。
【請求項３】導入計画が、アプリケーションの導入用のデータを含む応答ファイル・オブジェクトと、特定のワークステーション用の応答ファイル・オブジェクト・データをカストマイズするためのデータを含むカストマイズ・ファイル・オブジェクトとをさらに含むことを特徴とする、請求項２に記載の方法。
【請求項４】妥当性検査ステップが、導入計画オブジェクト内のデータとして指定されたターゲット・ワークステーション上のファイル・ディレクトリを検査して、そのファイル・ディレクトリが物理的に有効でネットワーク上でアクセス可能であるかどうかを判定することをさらに含み、ターゲット・ワークステーションが、そのアプリケーションを導入すべきワークステーションのグループのうちの１つであることを特徴とする、請求項２に記載の方法。
【請求項５】妥当性検査ステップが、ワークステーションのグループ上でのアプリケーションの導入に必要なコード・サーバ・ワークステーション上のオブジェクトの存在を確認することをさらに含み、コード・サーバ・オブジェクトが、導入計画オブジェクトの子オブジェクトではないことを特徴とする、請求項２に記載の方法。
【請求項６】妥当性検査ステップが、
アプリケーション・オブジェクトが、ネットワーク内の第１のアプリケーションに更新内容を提供するアプリケーションを表していることを判定するステップと、
アプリケーション・コンテナ・オブジェクト内の第１のアプリケーションを表す第１のアプリケーション・オブジェクトの存在を確認するステップとをさらに含むことを特徴とする、請求項２に記載の方法。
【請求項７】妥当性検査ステップが、ターゲット・ワークステーション上のオペレーティング・システムとその訂正サービスのレベルとを確認することをさらに含むことを特徴とする、請求項２に記載の方法。
【請求項８】導入計画オブジェクトが、導入すべき複数のアプリケーションと対応する応答ファイルとをそれぞれ表す、複数の計画中アプリケーション・オブジェクトと複数の対応する応答ファイル・オブジェクトとを含み、変換ステップが、ワークステーションのグループ内の特定のワークステーションに応じて複数の応答ファイル・オブジェクトをカストマイズするために、カストマイズ・ファイル・オブジェクトを複数の応答ファイル・オブジェクトに適用することをさらに含むことを特徴とする、請求項３に記載の方法。
【請求項９】複数の導入計画オブジェクトについて、事前妥当性検査ステップと、妥当性検査ステップと、変換ステップとを繰り返すステップと、
各ワークステーションごとにワークステーション待ち行列に複数の導入計画オブジェクトを待ち行列化するステップとをさらに含むことを特徴とする、請求項１に記載の方法。
【請求項１０】ネットワークでアプリケーションを導入するための導入計画オブジェクトをコミットするシステムにおいて、
導入計画オブジェクト内の子オブジェクトを検査し、必要であれば、導入計画オブジェクトに追加の子オブジェクトを追加することにより、導入計画オブジェクトを事前妥当性検査する手段と、
導入計画オブジェクトとその子オブジェクト内のデータを検査して、データにエラーがないかどうか確認することにより、導入計画オブジェクトを妥当性検査する手段と、
導入計画が正常に妥当性検査された場合に、導入計画オブジェクト内の子オブジェクトを、データ構造に応じてネットワークでアプリケーションを導入するネットワーク導入エンジンに使用可能なデータ構造に変換する手段とを含むシステム。
【請求項１１】導入計画オブジェクトが、アプリケーション・プログラムを表す計画中アプリケーション・オブジェクトと、アプリケーション・プログラムを導入すべきワークステーションのグループを表す計画中グループ・オブジェクトとを含むことを特徴とする、請求項１０に記載のシステム。
【請求項１２】導入計画が、アプリケーションの導入用のデータを含む応答ファイル・オブジェクトと、特定のワークステーション用の応答ファイル・オブジェクト・データをカストマイズするためのデータを含むカストマイズ・ファイル・オブジェクトとをさらに含むことを特

徴とする、請求項１１に記載のシステム。
【請求項１３】妥当性検査手段が、導入計画オブジェクト内のデータとして指定されたターゲット・ワークステーション上のファイル・ディレクトリを検査して、そのファイル・ディレクトリが物理的に有効でネットワーク上でアクセス可能であるかどうかを判定する手段をさらに含み、ターゲット・ワークステーションが、そのアプリケーションを導入すべきワークステーションのグループのうちの１つであることを特徴とする、請求項１１に記載のシステム。
【請求項１４】妥当性検査手段が、ワークステーションのグループ上でのアプリケーションの導入に必要なコード・サーバ・ワークステーション上のオブジェクトの存在を確認する手段をさらに含み、コード・サーバ・オブジェクトが、導入計画オブジェクトの子オブジェクトではないことを特徴とする、請求項１１に記載のシステム。
【請求項１５】妥当性検査手段が、
アプリケーション・オブジェクトが、ネットワーク内の第１のアプリケーションに更新内容を提供するアプリケーションを表していることを判定する手段と、
アプリケーション・コンテナ・オブジェクト内の第１のアプリケーションを表す第１のアプリケーション・オブジェクトの存在を確認する手段とをさらに含むことを特徴とする、請求項１１に記載のシステム。
【請求項１６】導入計画オブジェクトが、導入すべき複数のアプリケーションと対応する応答ファイルとをそれぞれ表す、複数の計画中アプリケーション・オブジェクトと複数の対応する応答ファイル・オブジェクトとを含み、変換ステップが、ワークステーションのグループ内の特定のワークステーションに応じて複数の応答ファイル・オブジェクトをカストマイズするために、カストマイズ・ファイル・オブジェクトを複数の応答ファイル・オブジェクトに適用することをさらに含むことを特徴とする、請求項１２に記載のシステム。
【請求項１７】複数の導入計画オブジェクトについて、事前妥当性検査ステップと、妥当性検査ステップと、変換ステップとを繰り返す手段と、
各ワークステーションごとにワークステーション待ち行列に複数の導入計画オブジェクトを待ち行列化する手段とをさらに含むことを特徴とする、請求項１０に記載のシステム。
【請求項１８】ネットワークでアプリケーションを導入するための導入計画オブジェクトをコミットするためにコンピュータで読取り可能な命令を格納するコンピュータ・メモリにおいて、
導入計画オブジェクト内の子オブジェクトを検査し、必要であれば、導入計画オブジェクトに追加の子オブジェクトを追加することにより、導入計画オブジェクトを事前妥当性検査する手段と、
導入計画オブジェクトとその子オブジェクト内のデータを検査して、データにエラーがないかどうか確認することにより、導入計画オブジェクトを妥当性検査する手段と、
導入計画が正常に妥当性検査された場合に、導入計画オブジェクト内の子オブジェクトを、データ構造に応じてネットワークでアプリケーションを導入するネットワーク導入エンジンに使用可能なデータ構造に変換する手段とを含み、
コンピュータ・メモリがコンピュータ・システムに結合され、コンピュータ・システムによってアクセスされたときに、その手段が活動化されることを特徴とするコンピュータ・メモリ。
【請求項１９】導入計画オブジェクトが、アプリケーション・プログラムを表す計画中アプリケーション・オブジェクトと、アプリケーション・プログラムを導入すべきワークステーションのグループを表す計画中グループ・オブジェクトとを含むことを特徴とする、請求項１８に記載のコンピュータ・メモリ。
【請求項２０】導入計画が、アプリケーションの導入用のデータを含む応答ファイル・オブジェクトと、特定のワークステーション用の応答ファイル・オブジェクト・データをカストマイズするためのデータを含むカストマイズ・ファイル・オブジェクトとをさらに含むことを特徴とする、請求項１９に記載のコンピュータ・メモリ。
【請求項２１】妥当性検査手段が、導入計画オブジェクト内のデータとして指定されたターゲット・ワークステーション上のファイル・ディレクトリを検査して、そのファイル・ディレクトリが物理的に有効でネットワーク上でアクセス可能であるかどうかを判定する手段をさらに含み、ターゲット・ワークステーションが、そのアプリケーションを導入すべきワークステーションのグループのうちの１つであることを特徴とする、請求項１９に記載のコンピュータ・メモリ。
【請求項２２】妥当性検査手段が、ワークステーションのグループ上でのアプリケーションの導入に必要なコード・サーバ・ワークステーション上のオブジェクトの存在を確認する手段をさらに含み、コード・サーバ・オブジェクトが、導入計画オブジェクトの子オブジェクトではないことを特徴とする、請求項１９に記載のコンピュータ・メモリ。
【請求項２３】妥当性検査手段が、アプリケーション・オブジェクトが、ネットワーク内の第１のアプリケーションに更新内容を提供するアプリケーションを表していることを判定する手段と、
アプリケーション・コンテナ・オブジェクト内の第１のアプリケーションを表す第１のアプリケーション・オブジェクトの存在を確認する手段とをさらに含むことを特徴とする、請求項１９に記載のコンピュータ・メモリ。
【請求項２４】導入計画オブジェクトが、導入すべき複

数のアプリケーションと対応する応答ファイルとをそれぞれ表す、複数の計画中アプリケーション・オブジェクトと複数の対応する応答ファイル・オブジェクトとを含み、変換ステップが、ワークステーションのグループ内の特定のワークステーションに応じて複数の応答ファイル・オブジェクトをカストマイズするために、カストマイズ・ファイル・オブジェクトを複数の応答ファイル・オブジェクトに適用することをさらに含むことを特徴とする、請求項２０に記載のコンピュータ・メモリ。
【請求項２５】複数の導入計画オブジェクトについて、事前妥当性検査ステップと、妥当性検査ステップと、変換ステップとを繰り返す手段と、
各ワークステーションごとにワークステーション待ち行列に複数の導入計画オブジェクトを待ち行列化する手段とをさらに含むことを特徴とする、請求項１８に記載のコンピュータ・メモリ。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、一般的には、コンピュータ・ネットワークでのソフトウェアの電子配布に関する。より具体的には、本発明はネットワーク上でネットワーク管理者が計画を実行する前にソフトウェアの導入に関する提案済み計画を妥当性検査（validating）するためのアーキテクチャに関する。
【０００２】
【従来の技術】ローカル・エリア・ネットワーク（ＬＡＮ）や広域ネットワーク（ＷＡＮ）などの計算機ネットワークを相互接続する複数のデータ処理システムを結合することは一般的なことである。それぞれのコンピュータ・システムは、複数のソフトウェア・プログラムを常駐させることになる。このようなネットワークの初期には、ネットワーク管理者が１組のディスケットを備えた各ワークステーションに１人ずつ向かい、導入、構成、または保守が必要なソフトウェア製品ごとに手作業でパラメータを入力していた。わずか数台のワークステーションであれば、複数の管理者とユーザがそれぞれのワークステーションに手作業でディスクを挿入し、導入または構成プロセス中の質問プロンプトに応答することができた。しかし、ネットワークのサイズが拡大するにつれ、当然のことながら、このような方法でソフトウェアを導入することがより難しくなってきた。また、ネットワークのサイズや複雑さ、ならびにコンピュータ・ソフトウェア製品間の相互依存度が増すにつれ、それぞれのワークステーションを構成することがますます複雑で時間を要し、エラーが発生しやすいプロセスになってきた。管理者は、もはやこの手作業による方法を行えなくなり、むしろ、ネットワークを介してワークステーションにアプリケーションを導入し構成するよう特別に設計された製品に転換している。
【０００３】したがって、リモートからコンピュータ・ソフトウェアを導入し、ワークステーションを構成することが、ますます、受け入れられるようになってきた。このプロセスについて一般に使用されている用語は、電子ソフトウェア配布（ＥＳＤ）である。ネットワーク内でのソフトウェアの電子配布は、ほぼ１０年間の間、当技術分野で知られてきた。より一般的なＥＳＤ製品としては、ＩＢＭのNetView分散管理プログラムおよびＬＡＮ構成導入分散ユーティリティや、Novell社のNetwork Navigatorなどがある。これらの製品は、ネットワークによるソフトウェアの配布、導入、構成という面倒な作業を軽減するよう設計されている。
【０００４】ＥＳＤ製品はすでに数年間存在しているが、ネットワーク管理者が直面するすべての問題に完全に対処するわけではない。ネットワーク管理者が直面する問題のうち未対応の問題の１つは、ネットワーク内の１群のワークステーション上での複数のソフトウェア・プログラムの配布、導入、構成を計画し実行することである。ソフトウェア製品間の相互依存度が増し、ネットワークが複雑であるため、これは気の重い作業である。
【０００５】これまで、ネットワーク管理者の負担を軽減するための努力が行われてきた。これまでの努力は、ネットワークによる画像の伝送方法や、導入および構成プロセスで必要になるファイルの物理的構築など、ネットワーク導入の「配管工事（plumbing）」に集中していた。これらの設計は、ネットワーク全体でのアプリケーションの伝送などの物理的作業の際に管理者にとって有用であったが、何が物理的作業であるかを計画する際には管理者に役立たなかった。
【０００６】本発明は、ネットワーク導入構成計画プロセスを一連の個別オブジェクトに分解するもので、この個別オブジェクトは、問題を抽出し、ＬＡＮ上の１組のワークステーションに導入し構成する予定のアプリケーションをセットアップし表示させるために管理者が使用する容易でオブジェクト指向の図形手段を提供するのに役立つものである。管理者は、このオブジェクト指向の表現を使用して、物理的な導入構成プロセスに必要なファイルを作成することができる。本発明は、管理者に対して問題を高レベルの表現で表示し、物理的な実施の多くから管理者を保護し、管理者が計画のためのブロックの構築に集中できるようにするものである。この設計を使用すると、管理者は、計画オブジェクトで妥当性検査し、物理的な導入および構成プロセスで必要な実際のファイルを作成するためにコミットすることができる、オブジェクトを「プレイグラウンド」すなわち非計画領域に作成することができる。
【０００７】本発明は、電子ソフトウェア配布の重要な改良策の１つを表している。
【０００８】
【発明が解決しようとする課題】したがって、本発明の一目的は、導入計画オブジェクトを妥当性検査すること

にある。
【０００９】本発明の他の目的は、サーバ・ディレクトリなどの前提ファイルを妥当性検査することにある。
【００１０】本発明の他の目的は、もともと計画から欠落していた前提オブジェクトまたは属性を導入計画に追加することにある。
【００１１】本発明の他の目的は、ＬＡＮ上のワークステーションでのアプリケーションの導入と構成のために必要なファイルを作成するために導入計画をコミットすることにある。
【００１２】本発明の他の目的は、オブジェクト指向ではなく、手続きベースのネットワーク導入ツールと結合することにある。
【００１３】
【課題を解決するための手段】上記およびその他の目的は、ネットワークでアプリケーションを導入するために導入計画オブジェクトをコミットすることによって達成される。導入計画オブジェクトは、アプリケーション・プログラムを表す計画中アプリケーション・オブジェクトと、そのアプリケーション・プログラムを導入するワークステーションのグループを表す計画中グループ・オブジェクトとを含む。コミット・プロセスの一部として、導入計画オブジェクトは、その子オブジェクトを検査し、必要であれば導入計画オブジェクトに追加の子オブジェクトを追加することによって事前妥当性検査され、導入計画オブジェクトとその子オブジェクトのデータにエラーがないかどうか検査することによって妥当性検査され、ネットワーク内でアプリケーションを導入するネットワーク導入エンジンに使用可能なデータ構造に変換される。そのアプリケーションの導入が応答ファイルと、特定のワークステーションに応じて応答ファイル・オブジェクト・データをカストマイズするためのデータを含むカストマイズ・ファイル・オブジェクトとを必要とする場合、導入計画は応答ファイル・オブジェクトをさらに含む。
【００１４】妥当性検査ステップでは、導入計画オブジェクト内のデータとして指定されたターゲット・ワークステーション上のファイル・ディレクトリを検査して、そのファイル・ディレクトリとファイル・オブジェクトが物理的に有効で、ネットワーク上でアクセス可能であるかどうかを判定する。また、妥当性検査ステップでは、ワークステーションのグループ上でのアプリケーションの導入に必要なオブジェクトがコード・サーバ・ワークステーション上に存在することも確認する。好ましい実施例では、コード・サーバ・オブジェクトは、導入計画オブジェクトの子オブジェクトではない。また、このステップでは、アプリケーション・オブジェクトが、ネットワーク内の第１のアプリケーションへの更新を行うアプリケーションを表すかどうかも判定する。このようなアプリケーションを表す場合、アプリケーション・コンテナ・オブジェクト内の第１のアプリケーションを表す第１のアプリケーション・オブジェクトが存在することを確認する。また、妥当性検査ステップでは、ターゲット・ワークステーション上のオペレーティング・システムとその訂正サービスのレベルも確認することができる。
【００１５】本発明の重要な特徴の１つは、導入計画オブジェクトが通常、複数の計画中アプリケーション・オブジェクトと、対応する複数の応答ファイル・オブジェクトとを含むことになる点である。これは、計画内のそれぞれのワークステーションに複数のアプリケーションを導入する予定で、対応する応答ファイルを１つずつ必要とするからである。ワークステーションのグループ内の特定のワークステーションに応じて複数の応答ファイル・オブジェクトをカストマイズするため、カストマイズ・ファイル・オブジェクトを複数の応答ファイル・オブジェクトに適用することができる。
【００１６】さらに、本発明は、複数の導入計画オブジェクトのために事前妥当性検査、妥当性検査、変換（transforming）の各ステップを繰り返すことも包含する。複数の導入計画オブジェクトは、それぞれのワークステーションごとに待ち行列化することができる。
【００１７】
【発明の実施の形態】本発明は、数種類のオペレーティング・システム下で様々なコンピュータ上またはコンピュータの集合上で動作することができる。このコンピュータとしては、たとえば、パーソナル・コンピュータ、ミニ・コンピュータ、メインフレーム・コンピュータ、他のコンピュータの分散ネットワーク内のコンピュータなどが可能である。特定のコンピュータの選択は、ディスクやディスク記憶域の要件のみに制限されるが、ＩＢＭ　ＰＳ／２（ＴＭ）コンピュータ・シリーズのコンピュータは本発明で使用できるはずである。ＩＢＭのＰＳ／２コンピュータ・シリーズの詳細については、Technical Reference Manual Personal Systems/2 Model 50, 60 Systems IBM Corporation（部品番号68X2224、資料番号S68X-2224）およびTechnical Reference 2 Manual Personal Systems/2 (Model 80) IBM Corporation（部品番号68X 2256、資料番号S68X-2254）を参照されたい。ＩＢＭ　ＰＳ／２パーソナル・コンピュータが動作可能なオペレーティング・システムの１つは、ＩＢＭのＯＳ／２　２．０（ＴＭ）である。ＩＢＭのＯＳ／２　２．０オペレーティング・システムの詳細については、OS/2 2.0 Technical Library, Programming Guide Vol. 1, 2, 3 Version 2.00（資料番号10G6261、10G6495、10G6494）を参照されたい。
【００１８】代替態様のコンピュータ・システムは、ＡＩＸ（ＴＭ）オペレーティング・システム上で動作するＩＢＭ　ＲＩＳＣシステム／６０００（ＴＭ）のコンピュータ系列に含まれるものと考えられる。ＲＩＳＣシス

テム／６０００の各種モデルについては、RISC System/6000, 7073 and 7016 POWERstation and POWERserverHardware Technical reference（資料番号SA23-2644-00）などのＩＢＭの数多くの資料に記載されている。ＡＩＸオペレーティング・システムについては、General Concepts and Procedure -- AIX Version 3 for RISC System/6000（資料番号SC23-2202-00）ならびにＩＢＭのその他の資料に記載されている。

【００１９】図１には、システム・ユニット１１と、キーボード１２と、マウス１３と、表示装置１４とを含むコンピュータ１０がブロック図形式で示されている。システム・ユニット１１は、１つまたは複数のシステム・バス２１を含み、そのバスに様々な構成要素が結合され、そのバスにより様々な構成要素間の通信が実施される。マイクロプロセッサ２２は、システム・バス２１に接続され、やはりシステム・バス２１に接続された読取り専用メモリ（ＲＯＭ）２３とランダム・アクセス・メモリ（ＲＡＭ）２４によってサポートされている。ＩＢＭ　ＰＳ／２シリーズのコンピュータのマイクロプロセッサは、３８６または４８６マイクロプロセッサを含むIntelのマイクロプロセッサ・ファミリーの１つである。しかし、この特定のコンピュータでは、６８０００、６８０２０、６８０３０マイクロプロセッサなどのMotorolaのマイクロプロセッサ・ファミリーや、ＩＢＭ製またはHewlett Packard、Sun、MotorolaなどのＰｏｗｅｒＰＣチップなどの様々な縮小命令セット・コンピュータ（ＲＩＳＣ）マイクロプロセッサを含み、かつこれに限定されないその他のマイクロプロセッサも使用可能である。

【００２０】ＲＯＭ２３は、各種コードのうち、対話、ディスク・ドライブ、キーボードなどの基本ハードウェア操作を制御する基本入出力システム（ＢＩＯＳ）を含んでいる。ＲＡＭ２４は、オペレーティング・システムとアプリケーション・プログラムがロードされるメイン・メモリである。メモリ管理チップ２５は、システム・バス２１に接続され、ＲＡＭ２４とハード・ディスク・ドライブ２６およびフロッピー・ディスク・ドライブ２７とのデータのやりとりを含む、直接メモリ・アクセス操作を制御する。やはりシステム・バス２１に結合されているＣＤ　ＲＯＭ３２は、マルチメディア・プログラムやプレゼンテーションなどの大量のデータを格納するために使用する。

【００２１】また、このシステム・バス２１には、キーボード制御装置２８、マウス制御装置２９、ビデオ制御装置３０、音声制御装置３１という様々な入出力制御装置も接続されている。予想できるように、キーボード制御装置２８はキーボード１２用のハードウェア・インタフェースを提供し、マウス制御装置２９はマウス１３用のハードウェア・インタフェースを提供し、ビデオ制御装置３０は表示装置１４用のハードウェア・インタフェースであり、音声制御装置３１はスピーカ１５ａおよび１５ｂ用のハードウェア・インタフェースである。トークン・リング・アダプタなどの入出力制御装置４０は、同様に構成された他のデータ処理システムへのネットワーク４６による通信を可能にするものである。

【００２２】本発明の好ましい実施態様の１つは、前述のように一般的に構成された１つまたは複数のコンピュータ・システムのランダム・アクセス・メモリ２４に常駐する命令セット４８～５２である。コンピュータ・システムが要求するまで、この命令セットは、ハード・ディスク・ドライブ２６、またはＣＤ－ＲＯＭ３２で最終的に使用される光ディスクやフロッピー・ディスク・ドライブ２７で最終的に使用されるフロッピー・ディスクなどの取外し可能メモリなど、他のコンピュータ・メモリに格納することができる。当業者であれば、命令セットの物理的記憶によって、それが電気的、磁気的、または化学的に格納される媒体が物理的に変化し、その結果、コンピュータが読取り可能な情報がその媒体に収容されることに留意されたい。命令、記号、文字などの用語で本発明を説明する方が便利であるが、これらの用語ならびに同様の用語はいずれも適切な物理要素に関連するものであることに留意されたい。また、オペレータに関連する可能性のある比較、妥当性検査、その他の用語で本発明を説明する場合も多い。ここに記載し、本発明の一部を形成するいずれの操作もオペレータのアクションは望まれない。これらの操作は、電気信号を処理して他の電気信号を生成する機械による操作を意味する。

【００２３】ワークステーションが統合されるネットワークとしては、ローカル・エリア・ネットワーク（ＬＡＮ）または広域ネットワーク（ＷＡＮ）があり、後者は既知のコンピュータ・アーキテクチャ下で動作する他のノードまたは複数システムのネットワークへのテレプロセシング接続を含む。いずれのノードでも、１つまたは複数の処理システムが存在し、それぞれが多かれ少なかれ前述のように構成された単一ユーザまたは複数ユーザ・システムである可能性がある。これらの処理システムは、サービスを要求する側か提供する側かに応じて、クライアントまたはサーバ・ワークステーションとして動作する。特定の実施態様では、本発明は、ＬＡＮサーバ４８、ＬＡＮ　ＣＩＤユーティリティ５０、および本発明が実施されているネットワーク導入アプリケーション５２を含む、ＩＢＭ　ＯＳ／２ＬＡＮサーバ・アーキテクチャによって相互接続された複数のＩＢＭ互換ワークステーション上で動作する。これらのアプリケーションは、一緒にパッケージ化されている場合もあれば、別々のアプリケーションとして販売される場合もある。ローカル・エリア・ネットワークに関する簡単な説明は、ラリー・Ｅ．ジョーダン（Larry E. Jordan）およびブルース・チャーチル（Bruce Churchill）著Communications and Networking For The IBM PC（Robert J. Brady発

行（A Prentice Hall Company）1983年）に記載されている。管理者にとって最も単純な構成は、ＬＡＮ上に１つのコード・サーバを設け、そのワークステーション上でネットワーク導入プログラムを実行する方法であると思われる。この設計は、管理者が複数のコード・サーバをＬＡＮ上で定義することを妨げるものではない。

【００２４】ＩＢＭによって定義されている構成、導入、分散（ＣＩＤ）の各種方法の１つは、応答ファイルの使用と、製品自体の導入プログラムとの組合せとに基づいている。ＣＩＤプロセスでは、ネットワーク管理者はネットワーク内の１台または複数台のマシンを、そこからネットワーク内の他のワークステーションが応答ファイルを含むコンピュータ・ソフトウェア・イメージを受け取るコード・サーバとして選択する。応答ファイルとは、導入または構成あるいはその両方のプロセス中にプログラムによって尋ねられる、対応する１組の質問に対する１組の応答を含むＡＳＣＩＩフラット・ファイルである。一般的なＣＩＤプロセスに関する詳細については、ＩＢＭの資料であるLAN Configuration Installation and Distribution Utility Guide（資料番号S10H-9742-00）を参照されたい。リダイレクトされた導入構成を自動化するプログラム・モジュールは、ＬＡＮ構成導入分配ユーティリティ（ＬＣＵ）と呼ばれる。以下に示す実施例の説明における詳細の多くは、ネットワーク内でアプリケーションを導入するという実際の物理的作業を実施するためのエンジンとしてＬＣＵを使用することによるものである。以下の好ましい実施例では、本発明のコミット・プロセスによって生成されるファイルは、ＬＣＵが必要とするコマンド・ファイルと応答ファイルである。これは、本発明のオブジェクト・ベースの設計を使用すると、ＬＣＵなどの手続き型導入エンジンにとって有用なファイルを作成できることを示している。他の導入エンジンを使用すると、コミット・プロセスにより、各種のファイルあるいは導入エンジンがオブジェクト指向であれば各種のオブジェクトが生成されるはずである。ＬＣＵコマンド・ファイルは、ネットワーク管理者ＷＫｓＡｐｐＩＰがワークステーション上での導入を計画している製品ならびにその製品を導入または構成すべき順序を識別する。管理者は、ネットワーク上の各ワークステーションごとに別々のＬＣＵコマンド・ファイルを用意することができる。応答ファイルには、そのワークステーションにソフトウェア製品を導入するのに必要な情報が含まれている。管理者は、アプリケーションごとに応答ファイルを１つずつ用意することができるが、その１つの応答ファイルは、アプリケーションの導入または構成プログラムによってサポートされている場合、追加の組込み応答ファイルを含むことができる。ワークステーション固有の情報が一切なければ、応答ファイルを複数のワークステーション間で共用することもできる。これらは同じ規則によって再導入または構成されるが、応答ファイルをアプリケーション間で共用することはできない。他に生成されるファイルとしては、コード・サーバの構成に使用する構成ファイルと、コード・ファイルへのアクセス権をクライアントに与えるために使用する権限リスト・ファイルがある。アプリケーション・イメージは、そのアプリケーションの導入元であるコード・サーバ上に格納されている。導入の現行状態を追跡するＬＡＮ　ＣＩＤユーティリティの手順により、各ステップが正しい順序で確実に実行される。当業者であれば、この環境が本発明を実施可能な数多くの環境の１つにすぎないことに留意されたい。

【００２５】本発明の重要な態様の１つは、導入計画オブジェクトとその副構成要素が１つのオブジェクト指向システム内で実現されることである。当然のことながら、ここに記載する実施例が示すように、オブジェクト指向の計画オブジェクトは、ＬＡＮ　ＣＩＤユーティリティなどの手続きベースの導入エンジンに移植することができる。オブジェクトやオブジェクト指向プログラミングは既知であるが、いくつかの面についてはここで言及する価値がある。それぞれのオブジェクトは、そのデータについて機能する所与のデータ属性とメソッドとを有する。データは、オブジェクトによって「カプセル化」されていると言われ、そのオブジェクトに属すメソッドによってのみ変更することができる。通常、メソッドは、所望のメソッドを識別し、必要な引数を供給するメッセージをオブジェクトに送ることによって呼び出される。新しいクラス・オブジェクトを作成するためにクラス・オブジェクトをサブクラス化することができる。「継承」とは、既存のオブジェクトからメソッドやデータ構造などのすべてのプロパティを継承する既存のオブジェクトから新しいオブジェクトを派生させる能力である。新しいオブジェクトは、既存のクラスの既存のメソッドに追加されるか、またはその既存のメソッドを指定変更する新しいメソッドなど、所与の固有の特徴を有する場合がある。新しいサブクラスでは、その既存の基本クラスからそれを区別するメソッドとデータを指定することだけが必要である。したがって、ソフトウェア開発者は、新しいコード全体を開発する必要はない。開発者は、そのソフトウェアの新しい固有の特徴を指定するだけでよい。オブジェクト指向の技術、概念、規則に関する背景情報については、グレーディー・ブーク（Grady Booch）著Object Oriented Design With Applications（The Benjamin/Cummins Publishing Company、1990年）およびB. マイヤー（Meyer）著Object Oriented Software Construction（Prentice Hall、1988年）を参照されたい。本発明の場合、導入計画およびその構成要素のオブジェクト指向性により、複数の計画オブジェクトまたはグループ・オブジェクト用のオブジェクトのインスタンスを作成することによって基本アプリケーションおよびワークステーション・オブジェクトの再使用が可

能になる。
【００２６】導入計画（ｐｌａｎ）オブジェクトと、それに収容される各種オブジェクトの階層表現を図２に示す。オブジェクトは、「プレイグラウンド」内の計画オブジェクトの外部に存在するか、またはそのオブジェクトの「計画中」インスタンスを作成する計画オブジェクトの内部に移動させることができる。計画オブジェクト２００は階層の最上部に位置し、管理者は一度に複数の計画オブジェクトを操作することができる。好ましい実施例によれば、有効な計画オブジェクトには、１つまたは複数の計画中アプリケーション（ＡｐｐＩＰ）オブジェクト２０２、２１０、２１４が収容されることになる。ＡｐｐＩＰオブジェクトは、ワークステーションのグループ上に導入または構成されるアプリケーション・プログラムを表している。また、有効な計画オブジェクトには、１つの計画中グループ（ＧｒｏｕｐＩＰ）オブジェクト２２０も収容されることになる。計画中グループ・オブジェクトはグループ・オブジェクトに似ているが、計画中ワークステーション・オブジェクトだけを保管し、ワークステーション・オブジェクトは保管しない。すべての計画中ワークステーション・オブジェクトが一義的である限り、１つの計画中グループが単一組のワークステーションまたはネットワーク全体を表すことも可能である。一義性は、ワークステーション識別名から同定される。本発明は、１つの導入計画で複数の計画中グループ・オブジェクトをサポートするように容易に拡張することができる。
【００２７】各ＡｐｐＩＰオブジェクトは、０または１個のカストマイズ・ファイル（ＣｕｓｔＦｉｌｅ）２０４と、０または１個の計画中カテゴリ（ＣａｔＩＰ）オブジェクト２０６、２１６、２２６、２３６とを収容することができる。また、各計画中アプリケーション・オブジェクトは、導入など、それが実行するアクションを１つ備え、その結果、計画中アプリケーションのアクションと一致する計画中カテゴリ・オブジェクトを１つだけ有することが好ましい。計画中アプリケーション・オブジェクトがアクションを実行するための応答ファイルを必要としない場合、計画中カテゴリも必要としない。ＣｕｓｔＦｉｌｅオブジェクトは、垂直軸にキーワード、水平軸にワークステーション名を備えたマトリックス・ファイルとして構成することができる。当業者は、他の構成も可能であることに留意されたい。たとえば、ＣｕｓｔＦｉｌｅオブジェクトは、表形式で格納されたインスタンス日付とともに応答ファイルを修正するのに必要なすべてのメソッドを含むことができる。コミット時には、１つまたは複数の計画中アプリケーション・ファイルにそのＣｕｓｔＦｉｌｅオブジェクトを適用し、特定の１組のワークステーションに応じてそれをカストマイズする。ＣａｔＩＰは、導入または構成など、計画中ワークステーション・オブジェクト上で実行可能なＡｐｐＩＰ用の指定アクションを表している。各ＣａｔＩＰオブジェクトは、０または１個の応答ファイル（ＲｓｐＦｉｌｅ）２０８、２１８、２２８、２３８を収容することになる。ＲｓｐＦｉｌｅは、アプリケーションを導入または構成するときに質問またはパネルへのオペレータ入力の代わりに計画中アプリケーション導入または構成プログラムが使用する、ＡＳＣＩＩキーワード等価値ファイルである。当業者は、ＧｒｏｕｐＩＰオブジェクトの下にＣｕｓｔＦｉｌｅオブジェクトも格納するようにこの実施例を拡張可能であることに留意されたい。
【００２８】ＧｒｏｕｐＩＰオブジェクトは、１つまたは複数の計画中ワークステーション（ＷｋｓＩＰ）オブジェクト２２２、２４０、２４４を収容することになる。各ＷｋｓＩＰオブジェクトは、１つまたは複数の計画中ワークステーション・アプリケーション（ＷｋｓＡｐｐＩＰ）オブジェクト２２４、２３０、２３４を収容することになる。各ＷｋｓＩＰオブジェクトごとのＷｋｓＡｐｐＩＰオブジェクトの数は、ＡｐｐＩＰオブジェクトの数に比例する。１つのワークステーション下の各ＷｋｓＡｐｐＩＰオブジェクトは、応答ファイルおよびその他の情報を伝播するためのＡｐｐＩＰオブジェクトへのリンクを有する。前述のように、ＷｋｓＡｐｐＩＰオブジェクトは、ＡｐｐＩＰオブジェクトの場合のように、０または１個のＣａｔＩＰオブジェクトも収容することができる。
【００２９】導入計画オブジェクトを作成するネットワーク導入プログラムは、１組の完全定義アプリケーションまたはＯＳ／２　Ｗａｒｐ™などのオペレーティング・システムとともに出荷される場合がある。このような構成には、導入および構成を実行するためにアプリケーションが必要とするすべての属性とコマンドが含まれている。管理者が追加のアプリケーションの導入と構成または既存のアプリケーションの導入と構成の修正を必要とする場合、まず、そのネットワーク内のワークステーションのグループと単独ワークステーションとを表す、ワークステーション・グループ・オブジェクトとワークステーション・オブジェクトとを作成し、計画オブジェクトを作成し、そのワークステーション・グループとアプリケーションを新たに作成した計画にドラッグすることから始めることができる。手作業でまたはカストマイズ・ファイルを使用して応答ファイルを修正する必要がない場合、管理者は、物理的な導入および構成プロセスに必要なファイルを作成するためにただちにその計画をコミットすることができる。様々な計画のためにアプリケーション・オブジェクトとワークステーション・オブジェクトを使用することは、本発明で特に重要な態様の１つである。他の計画で使用するためにこれらのオブジェクトをもう一度作成する必要はない。
【００３０】図３および図４は、導入計画内のオブジェ

クトに関連するが、導入計画内に存在しないオブジェクトを示している。図3のコード・サーバ・コンテナ・オブジェクト２５０は、１つまたは複数のコード・サーバ・オブジェクト２５２、２６０、２６４を含む。コード・サーバは、その他の情報のうち、ワークステーション上に導入または構成するアプリケーションのイメージを格納する。各コード・サーバ・オブジェクト２５２は、その製品イメージを表す、１つまたは複数のアプリケーション・イメージ（ＡｐｐＩｍａｇｅ）オブジェクト２５４および２５８を有することになる。１つのＡｐｐＩｍａｇｅオブジェクトは、アプリケーション・コンテナ・オブジェクト内の１つのアプリケーション・オブジェクトへのリンクを有する。リンクとは、ネットワーク導入プログラム内の他のオブジェクトを指し示す、オブジェクトの属性である。

【００３１】図４は、１つまたは複数のアプリケーション（Ａｐｐ）オブジェクト２７２、２８８、２９２を保管するアプリケーション・コンテナ・オブジェクト２７０を示している。各Ａｐｐオブジェクトは、１つまたは複数のカテゴリ（Ｃａｔ）オブジェクトを収容することができる。各Ｃａｔオブジェクトは、導入または構成など、Ａｐｐオブジェクトが実行可能なアクションに対応する。各Ｃａｔオブジェクトは、１つまたは複数のＲｓｐＦｉｌｅオブジェクトを有することができる。Ａｐｐオブジェクトが計画に追加されると、そのオブジェクトはＡｐｐＩＰオブジェクトに変換され、管理者が選択したアクション・タイプに対応するＣａｔオブジェクトだけがＣａｔＩＰオブジェクトに変換され、ＲｓｐＦｉｌｅがある場合はそれにデフォルトのマークが付けられる。デフォルト応答ファイルは、ほとんどの場合、管理者が導入または構成値を設定するために使用するものになるはずである。

【００３２】グラフィカル・ユーザ・インタフェースでは、ツリー・ビュー・ウィンドウ内のアイコンによるなど、同じ視覚的方法でＡｐｐオブジェクトとＡｐｐＩＰオブジェクトが管理者に示される。実施のために、様々なオブジェクトが計画オブジェクトと非計画オブジェクトに存在する。計画オブジェクトと非計画オブジェクトを表すために２つの互いに異なるが関連のあるオブジェクトを使用するか、またはそれが計画オブジェクトの一部であるかどうかを判定し、異なる属性を有するかまたは表示するためにそのクラスのｘｘｘｘｘインスタンスから呼び出せる属性フラグまたはメソッドとともに１つのクラスを使用することもできる。図５は、ネットワーク導入プログラムを提示するために可能なグラフィカル・インタフェースを示している。図５には計画オブジェクトのフォルダを示すオブジェクトのツリー・ビューが示されている。このフォルダは、「Plan Version 1」という計画オブジェクトの内容をほとんど示すようにオープンされている。「Plan Version 1」オブジェクトは、「OS/2 Warp Version 3」というタイトルのＡｐｐＩＰオブジェクトと、「Install - Category 7」というタイトルのＣａｔＩＰオブジェクトも含んでいる。また、計画オブジェクトは、「LAN Server 4.0 Entry- Requester」というタイトルの別のＡｐｐＩＰオブジェクトと、「Workstation Group 1」というタイトルのＧｒｏｕｐＩＰオブジェクトも含んでいる。このＧｒｏｕｐＩＰオブジェクトは、「Ted」、「John」、「Barbara」というタイトルのＷｋｓＩＰオブジェクトを示すように展開されている。「Barbara」というＷｋｓＩＰオブジェクトは、「OS/2 Warp Version 3」というタイトルのＷｋｓＡｐｐＩＰオブジェクトを示すように展開されており、そのオブジェクトは同じタイトルのＡｐｐＩＰオブジェクトと対応する。このＷｋｓＡｐｐＩＰオブジェクトは、「Install - Category 7」というタイトルのＣａｔＩＰオブジェクトを示すようにオープンされ、それは「F:＼nwi＼MODELRSP＼OS2W30＼INS＼WARPSAMP.RSP」というファイル名タイトルを備えた応答ファイルを含んでいる。もう一方のＷｋｓＡｐｐＩＰオブジェクトである「LAN Server 4.0 Entry - Requester」は図示していないが、これも「Barbara」ワークステーションの一部になるはずである。

【００３３】図１９の上の図は、ネットワーク導入プログラム用の最上位レベルのコンテナを示し、「Plan Construction」、「Status」、「Code Servers」、「Templates」の各オブジェクトを表示している。「Plan Construction」オブジェクトは、図１８のウィンドウに通じている。「Status」オブジェクトは、ワークステーションごとまたは計画ごとにアプリケーションの導入および構成の状況を管理者が表示できるようにするためのダイアログに通じている。「Code Servers」オブジェクトは、オープンすると、ネットワーク導入プログラムで定義されたコード・サーバ・オブジェクトを表示する。「Templates」オブジェクトは、オープンすると、管理者が計画またはアプリケーション・オブジェクトなどの正規オブジェクトを作成する際に使用するテンプレート・オブジェクトを提供する。オープンした「Templates」オブジェクトは図１９の下の図に示す。

【００３４】図５の図形表現は、ネットワーク導入プログラム内のオブジェクトを管理者に示すための唯一の方法ではない。たとえば、ＷｋｓＩＰの設定ノートブックで、ＷｋｓＡｐｐＩＰオブジェクトとその子オブジェクトを表示し、選択することも可能である。両方のオブジェクトが示されるわけではないが、ＡｐｐオブジェクトとａｐｐＩＰオブジェクトなどの同様のオブジェクトは、管理者が関連アイコンを物理的に変更しない限り、同じアイコン表現を有することになる。当業者であれば、ネットワーク導入プログラム用に代替グラフィカル・ユーザ・インタフェースを数多く設計できるはずである。

【００３５】導入計画内のオブジェクトを、導入計画用の物理的導入および構成プロセスが使用可能なファイルに変換するコミット・プロセスは、導入計画メニューからコミット・メニュー項目を選択することなどのアクションによって管理者がこの事象を開始することから始まる。このアクションにより、図７のメソッド入口ブロック４００で導入計画オブジェクトにCOMMITメソッドが送られる。コミット・プロセスはステップ４０５に移行し、そこで導入計画オブジェクトはコミット・プロセス中に検出される状況、警告、エラーの各メッセージを格納するためのエラー・ログ・オブジェクトERRORLOGを作成する。妥当性検査プロセス中に検出した重大エラーを追跡するために、VALERRORカウンタも設定される。

【００３６】次に、ステップ４１０で導入計画オブジェクトにPREVALIDATEメソッドが送られる。このPREVALIDATEメソッドは、導入または構成を正常に行うために必要な場合にオブジェクトまたは属性を計画オブジェクトに追加するもので、詳細については図８に示す。PREVALIDATEが復帰した後、４１５では、計画内に正しいデータがあるかどうかを検査するVALIDATEメソッドが導入計画オブジェクトに送られる。VALIDATEメソッドについては図９に示す。判断ブロック４２０では、VALERRORカウンタを検査する。VALERRORカウンタがゼロより大きい場合、処理はステップ４４５に移行し、そこでコミット・プロセス中に検出した警告およびエラーをERRORLOGが通知する。ログの通知後、COMMITメソッドは復帰ブロック４５０で終了する。

【００３７】VALERRORカウンタがゼロの場合、コミット処理はステップ４２５に移行し、そこで図１０に示すAPPLY CUSTOMIZATION FILESメソッドが導入計画オブジェクトに送られる。APPLY CUSTOMIZATION FILESメソッドから復帰後、導入計画オブジェクトはステップ４３０で計画出力ファイルを作成する。計画出力ファイルは、図２７に詳細を示し、ステップ４３５で呼び出されるINSTALL COMMAND SCRIPT GENERATIONルーチンへの媒介データとして機能する。INSTALL COMMAND SCRIPT GENERATIONルーチンの復帰後、導入計画は判断ブロック４４０で、コミット・プロセスの残りの部分で何らかのエラーが検出されたかどうかを検査する。エラーが検出された場合、処理はステップ４４５に移行する。エラーが一切検出されなかった場合、COMMITメソッドはブロック４５０で終了する。

【００３８】導入計画オブジェクト用のPREVALIDATEメソッドは、図８のメソッド入口ブロック５００から始まる。事前妥当性検査は、必要な場合にオブジェクトまたは属性を導入計画に追加するという点が妥当性検査とは異なる。妥当性検査では、プロセス中に後で問題を引き起こす可能性があるので管理者に警告する必要がある値とともに、正しい値または間違った値の有無だけを検査する。処理はステップ５０５に移行し、そこでMAINTENANCE SYSTEM REQUIRED FLAGとMAINTENANCE SYSTEM EXISTS FLAGの両方が偽に設定される。次のステップ５１０ではカウンタＩを１に設定し、判断ブロック５１５に移行し、そこでＩカウンタが導入計画内のＡｐｐＩＰオブジェクトの数以下であるかどうか確認するためにＩカウンタを検査する。この条件が真の場合、処理はステップ５３５に移行し、そこでカウンタＩによって索引付けされたＡｐｐＩＰオブジェクトにIS MAINTENANCE SYSTEM REQUIREDメソッドが送られる。このメソッドの復帰後、処理は判断ブロック５４０に移行し、真に設定されているかどうかを確認するためにMAINTENANCE SYSTEM EXISTS FLAGを検査する。これが真ではない場合、PREVALIDATEメソッドは残りのＡｐｐＩＰオブジェクトを検査する必要がなく、復帰ブロック５５０で終了することができる。MAINTENANCE SYSTEM EXISTS FLAGが真に設定されている場合、処理はステップ５４５に移行し、そこでＩカウンタを１だけ増分してから、判断ブロック５１５に移行する。

【００３９】判断ブロック５１５が偽であり、導入計画内のすべてのＡｐｐＩＰオブジェクトが検査された場合、処理は判断ブロック５２０に移行し、そこで真に設定されているかどうかを確認するためにMAINTENANCE SYSTEM REQUIRED FLAGを検査する。これが真である場合、PREVALIDATEメソッドは復帰ブロック５５０で終了することができる。MAINTENANCE SYSTEM REQUIRED FLAGが偽に設定されている場合、処理はステップ５２５に移行し、そこで導入計画オブジェクトは、TRANSPORTというアプリケーション・タイプおよびMAINTENANCE SYSTEMというアクション・タイプと同じ属性内の項目ＡｐｐＯｂｊｅｃｔから新しいＡｐｐＩＰオブジェクトを作成する。たとえば、ＯＳ／２構成では、ワークステーションのハード・ディスク上にあるＯＳ／２プログラムの最小バージョンが保守システム・アプリケーションになる。この最小ＯＳ／２は、ＯＳ／２プログラムのプレゼンテーション・マネージャ機能またはワークプレース・シェル機能を含まない。ＯＳ／２オペレーティング・システム用の導入アクションは、保守システムを必要とするようなプロセスである。次のステップ５３０では導入計画に新しいＡｐｐＩＰオブジェクトを追加し、PREVALIDATEメソッドは復帰ブロック５５０で終了する。

【００４０】導入計画オブジェクト用のVALIDATEメソッドは、図９のメソッド入口ブロック６００から始まる。処理は判断ブロック６０５に移行し、そこで導入計画オブジェクトは、導入計画が少なくとも１つのＡｐｐＩＰオブジェクトを含むかどうかを検査する。含まない場合、ステップ６３０でエラーをログ記録してから、処理が判断ブロック６３５に移行する。導入計画オブジェクトが少なくとも１つのＡｐｐＩＰオブジェクトを有する場合、ステップ６１０は、次にカウンタＩを１に設定することによって実行する。次に判断ブロック６１５が続

き、そこでＩカウンタが導入計画オブジェクト内のＡｐｐＩＰオブジェクトの数以下であるかどうかを確認するためにＩカウンタを検査する。この条件が真である場合、処理はステップ６２０に移行し、そこでカウンタＩによって索引付けされたＡｐｐＩＰオブジェクトにVALIDATEメソッドが送られる。このメソッドの復帰後、処理はステップ６２５に移行し、そこでＩカウンタを１だけ増分してから、判断ブロック６１５に移行する。

【００４１】判断ブロック６１５が偽であり、導入計画オブジェクト内のすべてのＡｐｐＩＰオブジェクトが図１１に示すようにそれぞれの妥当性検査メソッドを実行した場合、処理は判断ブロック６３５に移行し、そこで計画がＧｒｏｕｐＩＰオブジェクトを含むかどうかを確認するために導入計画オブジェクトが検査する。計画がそれを含まない場合、ステップ６４５でエラーをログ記録してから、処理は判断ブロック６５０に移行する。導入計画オブジェクトがＧｒｏｕｐＩＰオブジェクトを有する場合、ステップ６４０でVALIDATEメソッドがＧｒｏｕｐＩＰオブジェクトに送られ、その妥当性検査メソッドを実行する。

【００４２】判断ブロック６５０では、生成した応答ファイルの格納場所などのローカル・ディレクトリが正しく指定されているかどうか、またそのディレクトリが存在するかどうかを判定するために導入計画オブジェクトがそれ自体を検査する。ローカル・ディレクトリが存在しない場合、ステップ６５５でエラーがログ記録されてから、ステップ６６０に移行する。すべてのローカル・ディレクトリが指定され、存在している場合、処理はステップ６６０に移行し、そこでカウンタＩが１に設定される。次に判断ブロック６６５が続き、そこでＩカウンタが導入計画オブジェクト内に指定されたコード・サーバ・オブジェクトの数以下であるかどうかを確認するためにＩカウンタを検査する。この条件が真である場合、処理はステップ６７０に移行し、そこでカウンタＩによって索引付けされたコード・サーバ・オブジェクトにVALIDATEメソッドが送られる。このメソッドの復帰後、処理はステップ６７５に移行し、Ｉカウンタを１だけ増分してから、処理が判断ブロック６６５に移行する。判断ブロック６６５が偽であり、導入計画オブジェクト内のすべてのコード・サーバが検査された場合、VALIDATEメソッド用の処理は復帰ブロック６８０で完了する。

【００４３】導入オブジェクト計画用のAPPLY CUSTOMIZATION FILESメソッドは、図１０のメソッド入口ブロック７００から始まる。前述のように、このプロセスは、属性のように特定のワークステーション向けのアプリケーション用の応答ファイル内の値を修正するために管理者が使用する。すなわち、それぞれのワークステーションは、それに固有の何らかの値を有することができる。処理はステップ７０５に移行し、そこでカウンタＩが１に設定される。次に判断ブロック７１０が続き、そこでＩカウンタが導入計画に指定されたＡｐｐＩＰの数以下であるかどうかを確認するためにＩカウンタを検査する。この条件が偽である場合、APPLY CUSTOMIZATION FILESメソッドは復帰ブロック７８５で完了する。

【００４４】判断ブロック７１０ですべてのＡｐｐＩＰオブジェクトが検査されたわけではない場合、処理はステップ７１５に移行し、そこでＣｕｓｔＦｉｌｅオブジェクトを有するかどうかを判定するために、カウンタＩによって索引付けされたＡｐｐＩＰオブジェクトを検査する。ＣｕｓｔＦｉｌｅオブジェクトが検出されない場合、ステップ７３５ではカウンタＩを１だけ増分し、処理は判断ブロック７１０に移行する。ＣｕｓｔＦｉｌｅオブジェクトが検出された場合、ステップ７２０でＣｕｓｔＦｉｌｅが読み取られ、導入計画オブジェクトはＣｕｓｔＦｉｌｅオブジェクト内の指定のＷｋｓＩＰオブジェクトごとにキーワードと値の辞書を作成する。次に、ステップ７２５ではカウンタＪを１に設定し、判断ブロック７３０ではカウンタＪがＣｕｓｔＦｉｌｅに指定したＷｋｓＩＰオブジェクトの数以下であることを確認するために検査する。この条件が偽である場合、処理はステップ７８０に移行する。

【００４５】ＣｕｓｔＦｉｌｅオブジェクトに処理すべきＷｋｓＩＰオブジェクトが依然として存在する場合、ステップ７４０でカウンタＪを１に設定する。次に判断ブロック７４５では、カウンタＫがカウンタＪによって索引付けされたＷｋｓＩＰオブジェクトに属すＷｋｓＡｐｐＩＰオブジェクトの数以下であることを確認するために検査する。この条件が偽である場合、処理はステップ７８０に移行し、そこでカウンタＪを１だけ増分してから、次に判断ブロック７３０を実行する。

【００４６】カウンタＪによって索引付けされたＷｋｓＩＰオブジェクトに属すＷｋｓＡｐｐＩＰオブジェクトのすべてが検査されたわけではない場合、判断ブロック７５０では、カウンタＫによって索引付けされたＷｋｓＡｐｐ　２ｐがカウンタＩによって索引付けされたＡｐｐＺｐオブジェクトと同じアクション・タイプとアプリケーション・タイプを有するかどうかを確認するために検査する。ＸＸＸ高速処理はステップ７５５に移行し、そこでカウンタＫを１だけ増分し、判断ブロック７４５に移行する。フラグが一致である場合、処理はステップ７６０に移行し、そこでカウンタＫによって索引付けされたＷｋｓＡｐｐＩＰオブジェクトに属すＣａｔＩＰオブジェクトに対応するＲｓｐＦｉｌｅオブジェクトが取り出される。次に判断ブロック７６５では、ＲｓｐＦｉｌｅオブジェクトがＭＯＤＥＬタイプのものであるかどうかを確認するために検査する。これがＭＯＤＥＬタイプのものである場合、ステップ７７０でＲｓｐＦｉｌｅオブジェクトがＧＥＮＥＲＡＴＥＤタイプのＲｓｐＦｉｌｅオブジェクトに変換される。ＲｓｐＦｉｌｅがすでにＧＥＮＥＲＡＴＥＤタイプのものである場合、または

すでにＧＥＮＥＲＡＴＥＤタイプに変換されている場合、ステップ７７５では、カウンタＪによって索引付けされたＷｋｓＩＰオブジェクト用の辞書内のキーワード用の値に対応するファイル内の値を置換するために、ＲｓｐＦｉｌｅオブジェクト全体を探索することにより、ＣｕｓｔＦｉｌｅの変更内容をＲｓｐＦｉｌｅオブジェクトに適用する。その後、プロセスはステップ７５５に移行するはずである。

【００４７】計画中アプリケーション（ＡｐｐＩＰ）オブジェクト用のVALIDATEメソッドは図１１のメソッド入口ブロック８００から始まる。処理は判断ブロック８０５に移行し、そこでＡｐｐＩＰオブジェクトは指定のアクション・タイプ用にコマンド行が指定されているかどうかを確認するためにそのオブジェクト自体を検査する。たとえば、管理者がＡｐｐＩＰオブジェクトのアクション・タイプについてCONFIGUREを選択した場合、ＡｐｐＩＰオブジェクトにはCONFIGUREコマンド行ステートメントが指定されていなければならない。コマンド行が指定されていない場合、ステップ８１０でエラーをログ記録し、処理は判断ブロック８１５に移行する。

【００４８】そのアクション・タイプについてコマンド行が指定されている場合、処理は判断ブロック８１５に移行し、そこでＡｐｐＩＰオブジェクトはコード・サーバが指定されたかどうかを確認するために検査する。コード・サーバが指定されていない場合、ステップ８２０でエラーをログ記録し、処理は復帰ブロック８３０に移行する。コード・サーバが指定されている場合、ステップ８２５では指定のコード・サーバ・オブジェクトにDOES APPIMAGE EXIST FOR APPIPメソッドを送る。このメソッドからの復帰はＹＥＳまたはＮＯになる。復帰がＮＯである場合、ステップ８２０でエラーをログ記録する。メソッドからの復帰がＹＥＳである場合、処理は判断ブロック８３０に移行する。

【００４９】判断ブロック８３０では、ＡｐｐＩＰオブジェクトのアプリケーション・タイプがCORRECTIVE SERVICEであるかどうかを確認するために検査する。訂正サービス・アプリケーションとは、新しいＤＬＬファイルなどの更新モジュールをアプリケーションに適用するものである。更新モジュールは、問題箇所の修正や小規模な機能強化を含むことができる。アプリケーション・タイプがCORRECTIVE SERVICEではない場合、処理は判断ブロック８５０に移行する。アプリケーション・タイプがCORRECTIVE SERVICEである場合、判断ブロックでは、参照した訂正サービスＡｐｐオブジェクトがアプリケーション・コンテナ・オブジェクトに存在するかどうかを確認するために検査する。タイプがCORRECTIVE SERVICEではないＡｐｐまたはＡｐｐＩＰオブジェクトと照らし合わせて、訂正サービスＡｐｐオブジェクトまたはＡｐｐＩＰオブジェクトが適用される。訂正サービスＡｐｐが存在しない場合、ステップ８４０でエラーをログ記録し、処理は判断ブロック８５０に移行する。訂正サービスＡｐｐオブジェクトが存在する場合、ＡｐｐＩＰオブジェクトは訂正サービスＡｐｐオブジェクトにVALIDATEメソッドを送る。メソッドが復帰すると、処理は判断ブロック８５０に移行する。

【００５０】判断ブロック８５０では、ステップ８５５のＡｐｐＩＰオブジェクト用にＣａｔＩＰオブジェクトが存在するかどうかを確認するために検査する。ＣａｔＩＰオブジェクトが存在する場合、ＡｐｐＩＰオブジェクトはＣａｔＩＰオブジェクトにVALIDATEメソッドを送る。VALIDATEメソッドがＣａｔＩＰオブジェクトから復帰した場合、またはＡｐｐＩＰオブジェクト用にＣａｔＩＰオブジェクトが存在しなかった場合、処理は判断ブロック８６０に移行し、そこでＡｐｐＩＰオブジェクトはそれがＣｕｓｔＦｉｌｅオブジェクトを所有するかどうかを確認するために検査する。

【００５１】ＣｕｓｔＦｉｌｅオブジェクトがＡｐｐＩＰオブジェクトに属す場合、ステップ８６５でVALIDATEメソッドがＣｕｓｔＦｉｌｅオブジェクトに送られる。VALIDATEメソッドがＣｕｓｔＦｉｌｅから復帰した場合、またはＡｐｐＩＰオブジェクト用にＣｕｓｔＦｉｌｅが存在しなかった場合、ＡｐｐＩＰオブジェクト用のVALIDATEメソッド処理は復帰ブロック８７０で完了する。当業者は、使用する導入エンジンの要件に応じて、計画中アプリケーション・オブジェクト用の妥当性検査メソッドとして数多くの変形態様が存在することに留意されたい。

【００５２】ＡｐｐＩＰオブジェクト用のIS MAINTENANCE SYSTEM REQUIREDメソッドは図１２のメソッド入口ブロック９００から始まる。このメソッドはＡｐｐＩＰオブジェクトで必要になる。というのは、保守システム・アプリケーションが計画の一部である必要がある場合に、ＡｐｐＩＰオブジェクトは計画オブジェクトに応答する必要があるからである。処理は判断ブロック９０５に移行し、そこでＡｐｐＩＰオブジェクトは、ＡｐｐＩＰオブジェクトの指定のアクション・タイプについて保守システムが要求する属性が設定されているかどうかを確認するためにそのオブジェクト自体を検査する。属性が設定されている場合、ステップ８１０ではMAINTENANCE SYSTEM REQUIRED FLAGを真に設定し、処理は判断ブロック９１５に移行する。属性が設定されていない場合、判断ブロック９１５では次が実行され、ここでＡｐｐＩＰオブジェクトは、このアプリケーション・タイプがTRANSPORTに設定されているかどうか、ならびにそのアクション・タイプがMAINTENANCESYSTEM ONLYに設定されているかどうかを確認するために検査する。両方のタイプが一致する場合、ステップ９２０ではMAINTENANCE SYSTEM EXISTS FLAGを真に設定し、処理は復帰ブロック９２５で完了する。両方のタイプが一致しない場合、IS MAINTENANCE SYSTEM REQUIREDメソッドは復帰ブロック９２

５で完了する。
【００５３】ＡｐｐオブジェクトのVALIDATEメソッドは図１３のメソッド入口ブロック１０００から始まる。ＡｐｐオブジェクトとＡｐｐＩＰオブジェクトのVALIDATEメソッドは互いに異なる。というのは、Ａｐｐオブジェクトは、その子カテゴリ・オブジェクトの妥当性検査や、ＣｕｓｔＦｉｌｅオブジェクトのようにＡｐｐＩＰオブジェクトに属すがＡｐｐオブジェクトには属すことができないオブジェクトの妥当性検査を行う必要がないからである。処理は判断ブロック１００５に移行し、そこでＡｐｐオブジェクトは、指定のアクション・タイプ用にコマンド行が指定されているかどうかを確認するためにそのオブジェクト自体を検査する。コマンド行が指定されていない場合、ステップ１０１０でエラーをログ記録し、処理は判断ブロック１０１５に移行する。
【００５４】そのアクション・タイプについてコマンド行が指定されている場合、処理は判断ブロック１０１５に移行し、そこでＡｐｐオブジェクトは、コード・サーバが指定されているかどうかを確認するために検査する。コード・サーバが指定されていない場合、ステップ１０２０でエラーをログ記録し、処理は復帰ブロック１０３０に移行する。コード・サーバが指定されている場合、ステップ１０２５では指定のコード・サーバ・オブジェクトにDOES APPIMAGE EXISTメソッドを送る。このメソッドからの復帰はＹＥＳまたはＮＯになる。復帰がＮＯである場合、ステップ１０２０でエラーをログ記録する。メソッドからの復帰がＹＥＳである場合、VALIDATEメソッドは復帰ブロック１０３０で完了する。
【００５５】計画中ワークステーション（ＷｋｓＩＰ）オブジェクト用のVALIDATEメソッドは図１４のメソッド入口ブロック１１００から始まる。処理は判断ブロック１１０５に移行し、そこでＷｋｓＩＰオブジェクトは、オペレーティング・システムのＡｐｐオブジェクトが指定されているかどうかを確認するためにそのオブジェクト自体を検査する。オペレーティング・システムのＡｐｐオブジェクトが指定されている場合は、ＷｋｓＩＰオブジェクト上の導入済みオペレーティング・システムを管理者が把握し、そのオペレーティング・システムがそのオブジェクト上に設定されていることを意味し、処理はステップ１１１０に移行し、そこでオペレーティング・システムのＡｐｐオブジェクトにVALIDATEメソッドが送られる。メソッドの復帰後、判断ブロック１１１５では、ＷｋｓＩＰオブジェクト用に訂正サービス・アプリケーションが指定されているかどうかを確認するために検査する。訂正サービス・アプリケーションが指定されている場合、処理はステップ１１２０に移行し、そこで訂正サービスのＡｐｐオブジェクトにVALIDATEメソッドが送られる。VALIDATEメソッドが復帰すると、処理はステップ１１２５に移行する。訂正サービスのＡｐｐオブジェクトが指定されていない場合も、処理はステップ１１２５に移行する。
【００５６】オペレーティング・システムのＡｐｐオブジェクトが指定されていない場合も、処理はステップ１１２５に移行し、そこでカウンタＩが１に設定される。次に判断ブロック１１３０が続き、そこでＩカウンタがＷｋｓＩＰオブジェクト内のＷｋｓＡｐｐＩＰオブジェクトの数以下であるかどうかを確認するためにそのカウンタを検査する。この条件が真である場合、処理はステップ１１３５に移行し、そこでカウンタＩによって索引付けされたＷｋｓＡｐｐＩＰオブジェクトにVALIDATEメソッドが送られる。このメソッドの復帰後、処理はステップ１１４０に移行し、そこでＩカウンタを１だけ増分してから、判断ブロック１１３０に移行する。
【００５７】ＷｋｓＩＰオブジェクトに属するすべてのＷｋｓＡｐｐＩＰオブジェクトが検査された場合、VALIDATEメソッドは復帰ブロック１１４５で完了する。
【００５８】ＷｋｓＡｐｐＩＰオブジェクト用のVALIDATEメソッドは図１５のメソッド入口ブロック１２００から始まる。処理は判断ブロック１２０５に移行し、そこでＡｐｐＩＰオブジェクトは、それがＣａｔＩＰを含むかどうかを確認するためにそのオブジェクト自体を検査する。ＣａｔＩＰオブジェクトが検出されない場合、VALIDATEメソッドは復帰ブロック１２２０で終了する。ＣａｔＩＰオブジェクトが検出された場合、ＷｋｓＡｐｐＩＰオブジェクトはステップ１２１０でＣａｔＩＰオブジェクトにVALIDATEメソッドを送る。メソッドの復帰後、処理は復帰ブロック１２２０で終了する。
【００５９】ＣａｔＩＰオブジェクト用のVALIDATEメソッドは図１６のメソッド入口ブロック１３００から始まる。処理は判断ブロック１３０５に移行し、そこでＣａｔＩＰオブジェクトは、それがＲｓｐＦｉｌｅオブジェクトを含むかどうかを確認するためにそのオブジェクト自体を検査する。ＲｓｐＦｉｌｅオブジェクトが検出されない場合、VALIDATEメソッドは復帰ブロック１３２０で終了する。ＲｓｐＦｉｌｅオブジェクトが検出された場合、ＣａｔＩＰオブジェクトはステップ１３１０でＲｓｐＦｉｌｅオブジェクトにVALIDATEメソッドを送る。メソッドの復帰後、処理は復帰ブロック１３２０で終了する。
【００６０】ＲｓｐＦｉｌｅオブジェクトとＣｕｓｔＦｉｌｅオブジェクト用のVALIDATEメソッドは図１７のメソッド入口ブロック１４００から始まる。処理は判断ブロック１４０５に移行し、そこでそのファイル・オブジェクトは、その実ファイル名がローカルにアクセス可能なドライブ上のファイルとして物理的に存在するかどうかを確認するために検査する。ローカルまたはネットワークでアクセス可能なドライブ上でそのファイル名を検出できない場合、ステップ１４１０でエラーをログ記録し、処理は復帰ブロック１４１５に移行する。ファイル名が突き止められた場合、VALIDATEメソッドは復帰ブロ

ック１４１５で終了する。
【００６１】計画中グループ（ＧｒｏｕｐＩＰ）オブジェクト用のVALIDATEメソッドは図１８のメソッド入口ブロック１５００から始まる。処理はステップ１５１０に移行し、そこでカウンタＩを１に設定する。次に、判断ブロック１５１５では、ＩカウンタがＧｒｏｕｐＩＰオブジェクト内のＷｋｓＩＰオブジェクトの数以下であるかどうかを確認するために検査する。この条件が真である場合、処理はステップ１５１５に移行し、そこでＧｒｏｕｐＩＰオブジェクトはカウンタＩによって索引付けされたＷｋｓＩＰオブジェクトにVALIDATEメソッドを送る。このメソッドの復帰後、処理はステップ１５２０に移行し、そこでカウンタＩを１だけ増分し、制御は判断ブロック１５１０に戻る。判断ブロック１５１０が偽であって、ＧｒｏｕｐＩＰオブジェクト内のすべてのＷｋｓＩＰオブジェクトが検査された場合、VALIDATEメソッドは復帰ブロック１５２５で終了する。
【００６２】コード・サーバ・オブジェクト用のVALIDATEメソッドは図１９のメソッド入口ブロック１６００から始まる。処理は判断ブロック１６０５に移行し、そこでコード・サーバ・オブジェクトが実際に定義されており、別のオブジェクトで参照されるだけではないかどうかを確認するためにコード・サーバ・コンテナを検査する。コード・サーバが検出されない場合、ステップ１６１０でエラーをログ記録し、メソッドは復帰ブロック１６１５で完了する。
【００６３】コード・サーバ・オブジェクト用のDOES APPIMAGE EXIST FOR APPIPメソッドは図２０のメソッド入口ブロック１６３０から始まる。処理はステップ１６３５に移行し、そこでカウンタＩを１に設定する。次に、判断ブロック１６４０では、Ｉカウンタがコード・サーバ内のアプリケーション・イメージの数以下であるかどうかを確認するために検査する。この条件が真である場合、処理はステップ１６４５に移行し、そこでアプリケーション・イメージに属性として指定されたアプリケーション・インスタンス名をＡｐｐＩＰのインスタンス名と照らし合わせて検査する。２つのインスタンス名が一致する場合、DOES APPIMAGE EXIST FOR APPIPメソッドは終了し、ＹＥＳ（Ｙ）を返す。２つの名前が一致しない場合、ステップ１６５０でカウンタＩを１だけ増分し、処理は判断ブロック１６４０に移行する。判断ブロック１６４０が偽であって、コード・サーバ内のすべてのアプリケーション・イメージが検査された場合、DOES APPIMAGE EXIST FOR APPIPメソッドは終了し、ＮＯ（Ｎ）を返す。
【００６４】INSTALL COMMAND SCRIPT GENERATIONルーチンは、導入計画オブジェクトが作成した計画出力ファイル（ＰＯＦファイル）を読み取る。計画出力ファイルとその各セクションについては図２１に示す。計画セクション２００２は、応答ファイル、ログ・ファイル、ユーティリティ・ファイルに使用する計画名およびサーバ名など、その計画に固有のデータを含む。また、計画セクションは、１つのシーケンス・セクション２００４と、１つまたは複数のＡｐｐセクション２００６、ワークステーション・セクション２０１２、サーバ・セクション２０１８をも含む。シーケンス・セクション２００４は、ワークステーション上での計画実行中に実行する順序でそれぞれのアプリケーション／アクションをリストする。Ａｐｐセクション２００６は、短縮名、アクション、コマンド行、イメージ・サーバと別名、必要な環境など、計画内の各アプリケーションに固有のデータを含む。ワークステーション・セクション２０１２は、ワークステーション名、ブート・ドライブ、オペレーティング・システムの短縮名など、各ワークステーションに固有のデータを含む。その計画で参照される各サーバごとに、サーバ・セクションが１つずつ存在する。サーバ・セクション２０２２は、サーバ名と、応答ファイル、ログ・ファイル、ユーティリティ・ファイル用のその別名（マウント点）とを含む。導入計画オブジェクト・コードは数多くの様々な導入エンジンのために移植性が高くなっているので、単に導入エンジン用のファイルを直接生成するのではなく、計画出力ファイルを用意する方が便利である。以下に記載するコマンド・スクリプト生成ルーチンのみ、書き直す必要がある。
【００６５】INSTALL COMMAND SCRIPT GENERATIONルーチンは、計画内のワークステーション上で実行されるときに、グラフィカル・ユーザ・インタフェースで計画の進捗状況を示すためのベースとして機能する計画状況ファイルを作成する。計画状況ファイルとその各セクションについては図２２に示す。計画セクション２０３２はその計画に固有のデータを含む。ワークステーション・セクション２０３４は、計画内の各ワークステーションごとに１つずつ含まれる。各ワークステーション・セクションは、計画内の各アプリケーション・コマンド行に示される各ログ・ファイルごとにワークステーション・ログ２０３６を１つずつ含むことになる。また、各ワークステーション・セクションは、ワークステーション上の計画の全体的状況、たとえば、進行中、成功、失敗を含むファイルを指し示すワークステーション状況ファイル２０４２も１つずつ含むことになる。
【００６６】INSTALL COMMAND SCRIPT GENERATIONルーチンは、計画内の各ワークステーションごとに４つのファイルも作成する。この４つのファイルは、図２３に示すワークステーション待ち行列ファイルと、図２４のＰＲＥコマンド・ファイルと、図２５に示すＰＯＳＴコマンド・ファイルと、図２６に示すＬＣＵコマンド・ファイルである。
【００６７】図２３のワークステーション待ち行列ファイルは、ワークステーション上での実行を待っている計画２０６２の名前から構成される。このファイルは、タ

ーゲット・ワークステーション側で実行される複数の計画用の待ち行列化機構である別のプログラムNWICheckによって使用される。ターゲット・ワークステーション上で計画を実行する際に、このプログラムは待ち行列の最上部から実行済みの計画を除去する。この待ち行列化機構は、アプリケーション・プログラムをネットワークで導入する際の従来の労力に比べ、本発明が優れている重要な点の１つである。複数の計画オブジェクトを順次実行できるようにすることによって、複数のワークステーションでの複数のアプリケーションの導入を編成する作業を区分けし、その結果、かなり容易にすることができる。従来の労力では、除去しなければならない導入システムを一度に１つしか活動状態にすることができなかった。

【００６８】図２４に示すＰＲＥコマンド・ファイルは、ワークステーションがブート・ディスケットから計画実行を開始しない場合にワークステーションが計画を実行できるようにするためのものである。このＰＲＥコマンド・ファイルは、標準のコマンド言語で作成されている。これは、１つまたは複数の接続導入ステートメント２０７２と、１つのネットワーク導入使用可能性ステートメント２０７８とを含む。接続導入ステートメント２０７２は、計画実行中に必要になるリブート間に活動状態になるワークステーションにリダイレクトされたドライブを追加する。ネットワーク導入使用可能性ステートメント２０７８は、ワークステーションのリブート間に計画実行を容易にするような、必要なファイルまたはコマンドあるいはその両方を導入する。図２５に示すＰＯＳＴコマンド・ファイルは、リダイレクトされたドライブと、計画実行中またはＰＲＥコマンド・ファイルによってワークステーションに導入されたネットワーク導入使用可能性プログラムとを除去するためのものである。接続除去ステートメント２０８８は、リダイレクトされたドライブを除去する。ネットワーク導入使用可能ステートメント２０７８は、ネットワーク導入使用可能性プログラムによって事前に導入されたファイルまたはコマンドあるいはその両方を除去する。

【００６９】図２６に示すＬＣＵコマンド・ファイルは、ＬＡＮ　ＣＩＤユーティリティ・プログラムが必要とする構文で作成されたＲＥＸＸコマンド・ファイルであり、そのプログラムが必要とする変数を使用する。INSTALL COMMAND SCRIPT GENERATIONルーチンは、必要なリブートの回数を最小限に保ちながら、複数のフェーズで複数製品の導入を実行するのに必要なＲＥＸＸコードを動的に作成する。このコードは、ブート・ディスケットまたは他のハード・ディスク・ベースの方法により計画実行を開始するワークステーションを処理するために生成される。ＬＣＵコマンド・ファイルには７つのセクションがある。セットアップ・セクション２０９２は変数の初期設定から構成される。基本ＬＣＵ手続きセクション２１０４は、マシンのリブートや、ＣＩＤ戻りコードの処理などの事柄に使用する、様々なＲＥＸＸ手続きから構成される。INSTALL COMMAND SCRIPT GENERATIONルーチンが作成するＲＥＸＸコードは接続セクション２０９４から始まる。接続セクション２０９４は、ワークステーション上で計画を実行するのに必要な、リダイレクトされたすべてのドライブ用のリダイレクト・タイプ固有の接続（マウント）ステートメントを含む。アプリケーション・イメージ、ログ・ファイル、応答ファイル、その他の事柄すべてについて、接続が存在するものと思われる。変数セクション２０９６は、ブート・ドライブなど、ワークステーションに固有のいくつかの変数をセット・アップする。製品データ・セクション２０９８は、ＬＡＮ　ＣＩＤユーティリティが必要とする構文で作成された、計画内のアプリケーションごとのステートメントを含む。これらのステートメントとしては、とりわけ、アプリケーションおよびデフォルト応答ファイル用のコマンド行を含むものと思われる。導入セクション２１００は、アプリケーションの順序付け、エラー・アクション、リブート情報を含む。ＣｈｅｃｋＩＴ手続き２１０２は、エラー状態が発生したときと、導入の終了時に、導入セクション内から呼び出されるネットワーク導入手続きである。

【００７０】INSTALL COMMAND SCRIPT GENERATIONルーチンは図２７のブロック２２００から始まる。ステップ２２０５では、計画出力ファイル（ＰＯＦ）ファイル名やログ・ファイル名などのデータがないかどうか、コマンド行が解析される。処理はステップ２２１０に移行し、そこでコマンド行で渡されたＰＯＦファイルが読み込まれ、導入計画で参照された各コード・サーバごとにサーバ情報などの必要データがないかどうか解析される。ＰＯＦファイル構造については図２１に示す。

【００７１】次のステップ２２１５では、ターゲット・ディレクトリで計画が必要とするディレクトリ構造を作成する。このディレクトリ構造は、ワークステーション上で計画を実行するのに使用するツールによって予測される。当業者は、このディレクトリ構造が以下の好ましい実施例で記載するものとはかなり異なる可能性があることに留意されたい。ターゲット・ディレクトリには、その計画用のディレクトリが作成される。この計画ディレクトリから、その計画内の各アプリケーションごとにＬＣＵコマンド・ファイルと、ＰＲＥコマンド・ファイルと、ＰＯＳＴコマンド・ファイルと、ネットワーク導入状況ファイルとを含む、ログ・ファイル用のディレクトリが作成される。これらのファイルのそれぞれについては、ＬＡＮ　ＣＩＤユーティリティを使用して別々のディレクトリを作成しなければならない。というのは、これらのファイルはワークステーションによって命名され、同じディレクトリに同じ名前の２つの個別ファイルが常駐することはできないからである。

【００７２】処理はステップ２２２０に移行し、そこで計画状況ファイルの計画セクションが書き込まれる。計画状況ファイルの構造については図２２に示す。次にステップ２２２５では、サーバ／別名ドライブ・テーブルが構築される。このテーブルは、ターゲット・ワークステーションで計画を実行する際に使用するサーバとそれに対応する別名（マウント点）のすべてを含む。サーバ上のマウント点とは、ドライブ文字としてアプリケーションを接続できる論理位置である。サーバ／別名のすべての組合せをテーブルに追加した後、ステップ２２３０で重複部分が除去され、固有のサーバ／別名の組合せにドライブが割り当てられる。ドライブＺ：は、ネットワーク導入ユーティリティの別名になるドライブに割り当てられる。ドライブＹ：は、オペレーティング・システム固有のファイルに接続されるドライブに割り当てられる。固有のサーバ／別名の組合せの残りには、ドライブＷ：から逆の英字順にドライブ文字が割り当てられる。他のデバイス割当ても可能であるが、ターゲット・ワークステーションに物理的に接続されたデバイスが使用する既存のドライブ文字との競合を避けるために、ドライブが逆の英字順に割り当てられる。

【００７３】ステップ２２４０では、計画の実行中に導入すべきアプリケーションの数が決定される。これは、ＰＯＦファイルで定義されたアプリケーションの数に２を加えたものとして定義される。ネットワーク導入使用可能性プログラム用に１を加え、何らかの場合に導入される一時リダイレクタ用に１を加える。

【００７４】次に、INSTALL COMMAND SCRIPT GENERATIONルーチンは、ＰＯＦファイルで定義されたすべてのワークステーションについて、ステップ２２４５～２２８０のループになる。ループ内の第１のステップ２２５０では、現行計画の名前をワークステーションの待ち行列ファイルに付加する。ワークステーション待ち行列ファイルの構造については図２３に示す。次のステップ２２５５では、アプリケーションのコマンド行で変数置換を実行する。ソース・ディレクトリ、ログ・ファイル、クライアント名などのデータはそれぞれの計画またはワークステーションあるいはその両方に固有のものなので、システムに対してアプリケーションを定義するときに管理者が修正するのではなく、計画コミット時に置換しなければならない。次にステップ２２６０は、計画状況ファイル上の現行ワークステーション用のワークステーション・セクションである。計画状況ファイルについては図２２に示す。各アプリケーション・コマンド行上の／１：パラメータによって指定された各ログ・ファイルとともに、ワークステーションの状況ファイルのサーバに対するローカル経路が計画状況ファイルにリストされる。

【００７５】次に、ステップ２２６５では、プロセスはワークステーションのＰＲＥコマンド・ファイルを作成する。ＰＲＥコマンド・ファイルの構造については図２４に示す。ステップ２２３０で決定したネットワーク導入ユーティリティの別名用として、接続導入使用可能性ステートメントが追加される。接続導入ステートメントとは、オペレーティング・システムのスタートアップ・ファイルと、ファイル・サーバへの接続ステートメントにステートメントを導入するプログラムである。また、必要であれば、オペレーティング・システム固有のファイル用としても接続導入ステートメントが追加される。また、現行ワークステーション用の適切なネットワーク導入ステートメントも追加される。ステップ２２７０では、ＰＯＳＴコマンド・ファイルが作成される。ＰＯＳＴコマンド・ファイルの構造については図２５に示す。ＰＲＥコマンド・ファイル（またはＬＣＵコマンド・ファイル内の等価物）で実行したことを基本的に「取り消す」ためにステートメントが追加される。

【００７６】次にINSTALL COMMAND SCRIPT GENERATIONルーチンはステップ２２７５でCREATE LCU COMMAND FILEルーチンを呼び出す。このCREATE LCU COMMAND FILEルーチンについては図２８に示す。次のステップでは、次のワークステーションに進んで、２２４５の判断ブロックに移行する。処理すべき追加のワークステーションがまだ存在する場合はステップ２２５０～２２８０を繰り返し、存在しない場合はステップ２２８５でINSTALL COMMAND SCRIPT GENERATIONルーチンが計画状況ファイルをクローズし、呼出し側プログラムに戻る。

【００７７】CREATE LCU COMMAND FILEルーチンは図２８のブロック２３５０から始まる。ＬＣＵコマンド・ファイルの構造については図２６に示す。ＬＣＵコマンド・ファイルを作成する際の第１のステップは、ステップ２３５５で、ＬＣＵコマンド・ファイル用の事前定義済みセットアップ・セクションを書き出すことである。セットアップ・セクションはＬＡＮ　ＣＩＤユーティリティに固有のものであるが、他の導入エンジンを使用した場合でも、同様のセクションがこれらのファイルに共通のものになる可能性がある。ステップ２３６０では、INSTALL COMMANDSCRIPT GENERATIONルーチンのステップ２２３０で決定した固有のサーバ／別名対ごとのドライブ用にネットワーク固有の接続ステートメントが書き込まれる。ステップ２３６５～２３８０では、CREATE LCU COMMAND FILEルーチンが、WRITEVARIABLES SECTIONルーチン（図２９に示す）と、WRITE PRODUCT DATA SECTIONルーチン（図３０に示す）と、WRITE INSTALL SECTIONルーチン（図３２に示す）と、WRITE CHECKIT PROCEDUREルーチン（図３１に示す）とを呼び出す。ステップ２３８５では、ＬＣＵコマンド・ファイルのために上記のステップ２３５５でコピーした事前定義済み基本手続きがワークステーションのＬＣＵコマンド・ファイルに書き込まれる。次に、CREATE LCU COMMAND FILEルーチンは呼出し側の手続きに戻る。

【００７８】WRITE VARIABLES SECTIONルーチンは図２９のステップ２４５０から始まる。このルーチンは、ワークステーションに固有の変数を書き出す。まず、ステップ２４５５で、プロセスはワークステーションのブート・ドライブに対応するＲＥＸＸ変数を書き込む。ＲＥＸＸ構文では、これは、たとえばbootdrive=D:となるはずである。次に、ステップ２４６０では、ワークステーションでの計画実行中にＬＣＵが使用する事前定義済み変数がこのファイルに書き込まれる。次に、WRITE VARIABLES SECTIONルーチンは、復帰ブロック２４６５で呼出し側手続きに戻る。

【００７９】WRITE PRODUCT DATA SECTIONルーチンは図３０のステップ２５００から始まる。製品データ・セクションの各構成要素については図２６のボックス２０９８に説明されている。ステップ２５０５では、ネットワーク導入ユーティリティへの接続に必要な製品データがこのセクションに書き込まれる。これは、INSTALL COMMAND SCRIPT GENERATIONルーチンのステップ２２６０で説明したように、ＰＲＥコマンド・ファイルに書き込まれたものと同じになるはずである。必要であれば、オペレーティング・システム固有のファイルへの接続に必要な製品データもこのステップで書き込まれるはずである。ステップ２５１０では、ネットワーク導入使用可能性プログラム用の製品データが書き込まれる。ネットワーク導入ユーティリティは導入システムを実行するもので、ネットワーク導入使用可能性プログラムはネットワーク導入ユーティリティを導入するものである。ステップ２５１５～２５２５のループでは、計画で導入すべき各アプリケーションごとに製品データを書き出す。すべての製品データが書き出された後で、コマンド・ファイルにいくつの製品データ定義が含まれているかを示す１行が書き出される。この数は、INSTALL COMMAND SCRIPT GENERATIONルーチンのステップ２２４０で決定したものである。次に、WRITE PRODUCT DATA SECTIONルーチンは呼出し側手続きに戻る。

【００８０】WRITE CHECKIT PROCEDUREルーチンは図３１のステップ２６００から始まる。ＣｈｅｃｋＩｔ手続きは、ＬＣＵ　ＲＥＸＸコマンド・ファイルに追加される手続きである。ＣｈｅｃｋＩｔ手続きは、アプリケーションの導入が失敗に終わるか、計画が正常に完了したときに必ず呼び出されるものである。ＣｈｅｃｋＩｔ手続きは、別のプログラムを呼び出して、それにワークステーションの名前と計画の状況（成功または失敗）を渡す。この呼び出されるプログラムNWICHECKは、計画実行中にワークステーションの待ち行列ファイルと状況ファイルの保守を行う。まず、ステップ２６０５では、ＣｈｅｃｋＩｔ手続きの事前定義済みの開始が書き込まれる。次に、ステップ２６１０で、NWICHECKコマンド行での変数置換が行われ、ワークステーション名などのデータとログ・ファイルを置換する。ステップ２６１０で生成したNWICHECKステートメントはステップ２６１５でコマンド・ファイルに書き込まれる。ステップ２６２０でＣｈｅｃｋＩｔ手続きの残りが付加される。次に、WRITE CHECKIT PROCEDUREルーチンは呼出し側手続きに戻る。

【００８１】WRITE INSTALL SECTIONルーチンは図３２のステップ２７００から始まる。導入セクションの各構成要素については図２６のボックス２１００に説明されている。第１のステップ２７０５では、状態を０に設定する。次に、ステップ２７１０では、ＲＥＸＸを書き出すことによりループが構築される。次に、WRITE INSTALL SECTIONルーチンは、判断ブロック２７１５から計画内のすべてのアプリケーションを通るループになる。

【００８２】判断ブロック２７１５では、これが処理中の第１のアプリケーションであるかどうかをテストし、第１のアプリケーションではない場合はステップ２７２０で状態が１だけ増分され、第１のアプリケーションである場合は処理が判断ブロック２７２５に移行する。

【００８３】判断ブロック２７２５では、アプリケーションが保守システムを必要とするかどうかを検査する。必要とする場合、処理はステップ２７６０に移行し、そこでMAINT SECTIONルーチン（図３７）が呼び出される。ステップ２７６５で状態が１だけ増分され、次にステップ２７７０でWRITE APPルーチン（図３３）が呼び出される。次に、処理は判断ブロック２７４５に移行する。

【００８４】判断ブロック２７２５で保守システムが必要ではなかった場合、処理は判断ブロック２７３０に移行し、そこで導入中にプレゼンテーション・マネージャ（ＰＭ）が活動状態になることをそのアプリケーションが要求しているかどうかが検査される。プレゼンテーション・マネージャは、グラフィカル・ユーザ・インタフェース自体を管理する必要なしに、アプリケーションがユーザにグラフィカル・ユーザ・インタフェースを提示できるようにするものである。ＰＭが必要な場合、処理はステップ２７７５に移行し、そこでHARD DISKルーチン（図３９）が呼び出される。次に、ステップ２７８０で状態が１だけ増分され、処理は判断ブロック２７４５に移行する。判断ブロック２７３０でＰＭが必要ではなかった場合、処理はステップ２７３５に移行し、そこでWRITE APPルーチン（図３３）が呼び出される。処理は判断ブロック２７４０に移行し、そこでこれが処理中の第１のアプリケーションである場合、ステップ２７８５でBOOT DISKルーチン（図３８）が呼び出され、そうではない場合、処理は判断ブロック２７４５に移行する。

【００８５】判断ブロック２７４５で最後のアプリケーションが処理中ではない場合、処理はステップ２７５０に移行し、そこでEND SECTIONルーチン（図３６）が呼び出される。次に、処理は判断ブロック２７１５に戻る。判断ブロック２７４５で最後のアプリケーションが

処理中である場合、処理はステップ２７８５に移行し、そこでLAST APPルーチンが呼び出される。次に、WRITE INSTALL SECTIONルーチンは呼出し側手続きに戻る。

【００８６】WRITE APPルーチンは図３３のステップ２８５０から始まる。このルーチンは、コマンド・ファイル内の１つの全体状態について１つまたは複数のＲｕｎＩｎｓｔａｌｌステートメントを書き出す。ステップ２８５５では、選択ステートメントの適当なセクションを示すＲＥＸＸ構文が書き出される。次に、ステップ２８６０では、処理中のアプリケーションについてWRITE RUNINSTALLルーチン（図３４）が呼び出される。このルーチンは、コマンド・ファイル内の１つのＲｕｎＩｎｓｔａｌｌステートメントの論理を書き出す。処理は判断ブロック２８６５に移行し、そこで処理中のアプリケーションがオペレーティング・システムではない場合、ルーチンは呼出し側手続きに戻る。オペレーティング・システムである場合、処理はステップ２８７０に移行し、そこで計画で定義されたトランスポート・アプリケーション用にWRITE RUNINSTALLルーチンが呼び出される。次に、ステップ２８７５では、計画内のリダイレクタ用にWRITE RUNINSTALLルーチンが呼び出される。ステップ２８８０では、ネットワーク導入使用可能性プログラム用にWRITE RUNINSTALLルーチンが呼び出される。次に、復帰ブロック２８８５でWRITE APPルーチンは呼出し側手続きに戻る。

【００８７】WRITE RUNINSTALLルーチンは図３４のステップ２９００から始まる。ステップ２９０５と２９１０では、ＲＥＸＸ構文が次のようにコマンド・ファイルに書き出される。

【００８８】

```
If RunInstall(x) == BAD_RC then
    Call Checkit('STOP')
```

【００８９】この場合、xはこのルーチンの呼出しの対象となるアプリケーションのアプリケーション短縮名である。次に、WRITE RUNINSTALLルーチンは呼出し側手続きに戻る。

【００９０】LAST APPルーチンは図３５のステップ２９５０から始まる。このルーチンは、ＬＣＵコマンド・ファイルの導入セクションを終了するために使用するＲＥＸＸロジックを書き込む。ステップ２９５５では、ＲＥＸＸ構文が次のように書き込まれる。

【００９１】

```
If CALL_AGAIN = 0 then do
  CheckIt('SWITCH)
  call reboot
 end
 else
    call checkboot
 end
otherwise nop
end
end
```

【００９２】このコードは、ＬＣＵ変数であるCALL-AGAINが設定されているかどうかを検査するもので、この変数が設定されていない場合は、（１つ存在すれば）ワークステーション待ち行列内の次の計画に切り替えるためにＣｈｅｃｋＩＴルーチンが呼び出される。現行の全体状態になっているいずれかのプログラムがreboot-and-Callback戻りコードを返す場合は、CALL-AGAIN変数がＬＣＵによって１に設定される。call-againが１に設定されている場合、ワークステーションをリブートするためのＬＣＵ手続きが呼び出される。次に、LAST APPルーチンは呼出し側手続きに戻る。

【００９３】END SECTIONルーチンは図３６のステップ３０００から始まる。このルーチンはＬＣＵコマンド・ファイル内の１つの全体状態を終了するために使用するＲＥＸＸロジックを書き込む。ステップ３００５と３０１０では、ＲＥＸＸ構文が次のように書き出される。

【００９４】

```
call checkboot
end
```

【００９５】このコードは、ワークステーションの条件付きリブートのためのＬＣＵ手続きを呼び出す。次に、END SECTIONルーチンは呼出し側手続きに戻る。

【００９６】MAINT SECTIONルーチンは図３７のステップ３１００から始まる。このルーチンは、ターゲット・ワークステーション上に保守システムを導入するのに必要なＲＥＸＸロジックとＲｕｎＩｎｓｔａｌｌステートメントを書き込む。ステップ３１０５では、選択ステートメントの適当なセクションを示すＲＥＸＸ構文が書き出される。次にステップ３１１０で、ブート・ドライブがディスケットであるかどうかを検査するための構文が書き出される。ステップ３１１５では、計画で定義されたオペレーティング・システムの保守バージョン用にWRITE RUNINSTALLルーチンが呼び出される。ステップ３１２０では、計画で定義されたトランスポート・アプリケーションの保守バージョン用にWRITE RUNINSTALLルーチンが呼び出される。ステップ３１２５では計画内のリダイレクタ用にWRITE RUNINSTALLルーチンが呼び出され、ステップ３１３０ではネットワーク導入使用可能性プログラム用にWRITE RUNINSTALLルーチンが呼び出される。次に、ステップ３１３５では、END SECTIONルーチンが

呼び出される。次に、MAINT SECTIONルーチンは呼出し側手続きに戻る。

【００９７】BOOT DISKルーチンは図３８のステップ３２００から始まる。このルーチンは、ブート・ディスケットから導入を開始したときにターゲット・ワークステーションのハードファイルにネットワーク導入ユーティリティと必須リダイレクタを導入するのに必要なＲＥＸＸロジックとｒｕｎＩｎｓｔａｌｌステートメントを書き込む。第１のステップは、これが最後のアプリケーションであるかどうかを判定する判断ブロック３２０５である。これが最後のアプリケーションである場合、ＬＣＵコマンド・ファイルに次のようにＲＥＸＸ構文を書き出してから、ステップ３２１５に移行する。

【００９８】if CALL_AGAIN = 1 then do

【００９９】これが最後のアプリケーションである場合はCall-AGAIN変数が検査される。というのは、このアプリケーションがコール・バックを必要としない場合、ハード・ファイル上にコール・バック・ユーティリティを乗せる必要がないからである。この検査は本質的にはリブートを節約する。これが最後のアプリケーションではない場合、処理はステップ３２１５に移行し、そこでブート・ドライブがディスケットであるかどうかを検査するための構文が書き出される。ステップ３２２０では計画内のリダイレクタ用にWRITE RUNINSTALLルーチンが呼び出され、ステップ３２２５ではネットワーク導入使用可能性プログラム用にWRITE RUNINSTALLルーチンが呼び出される。ステップ３２３０で終了が書き込まれ、処理は判断ブロック３２３５に移行し、そこでこれが最後のアプリケーションであるかどうかを検査する。これが最後のアプリケーションではない場合、BOOT DISKルーチンは呼出し側手続きに戻り、最後のアプリケーションである場合は、ステップ３２４０でＲＥＸＸ構文が次のように書き込まれてから、呼出し側手続きに戻る。

```
Do forever
  Select
   when OVERALL_STATE = 0 then do
     if RunInstall(transport) == BAD_RC then
       call CheckIt('STOP')
     if IsBootDriveRemovable = 1 then do
       if RunInstall(redirector) == BAD_RC then
         call CheckIt('STOP')
       if RunInstall(ni_enable) == BAD_RC then
         call CheckIt('STOP')
         end
         call checkboot
       end
       when OVERALL_STATE = 1 then do
         if RunInstall(os_maint) == BAD_RC then
           call CheckIt('STOP')
         if RunInstall(xport_maint) == BAD_RC then
```

【０１００】

```
call checkboot
end
```

【０１０１】このコードは、現行の全体状態で事前に導入されたプログラムからのＣＩＤ戻りコードに基づいて、マシンの条件付きリブートを行うためのＬＣＵ手続きを呼び出すものである。

【０１０２】HARD DISKルーチンは図３９のステップ３３００から始まる。このルーチンは、導入中のアプリケーションがＰＭを必要とする場合にブート・ディスケットからブートしたときに、ターゲット・ワークステーションのハード・ファイルにネットワーク導入ユーティリティと必須リダイレクタを導入するのに必要なＲＥＸＸロジックとＲｕｎＩｎｓｔａｌｌステートメントを書き込む。ステップ３３０５では、選択ステートメントの適当なセクションを示すＲＥＸＸ構文が書き出される。次に、ステップ３３１０では、ブート・ドライブが固定ドライブであるかどうかを検査するための構文がＬＣＵコマンド・ファイルに書き出される。ステップ３３１５では、計画内のリダイレクタ用にWRITE RUNINSTALLルーチンが呼び出される。ステップ３３２０では、ネットワーク導入使用可能性プログラム用にWRITE RUNINSTALLルーチンが呼び出される。ステップ３３２５ではEND SECTIONルーチンを呼び出す。次に、HARD DISKルーチンは呼出し側手続きに戻る。保守システムまたはプレゼンテーション・マネージャを必要としないトランスポートと、保守システムを必要とするオペレーティング・システム・サービスと、プレゼンテーション・マネージャを必要とするアプリケーションを順番に導入する計画によってWRITE INSTALL SECTIONルーチンを実行後、次のような導入セクションで終わるはずである。

【０１０３】

```
 call CheckIt('STOP')
 if RunInstall(redirector) == BAD_RC then
   call CheckIt('STOP')
 if RunInstall(ni_enable) == BAD_RC then
 call CheckIt('STOP')
 call checkboot
end
when OVERALL_STATE = 2 then do
  if RunInstall(os_service) == BAD_RC then
    call CheckIt('STOP')
  call checkboot
end
when OVERALL_STATE = 3 then do
if RunInstall(app) == BAD_RC then
  call CheckIt('STOP')
if CALL_AGAIN = 0 then do
  call CheckIt('SWITCH')
  call Reboot
 end
 else call checkboot
end
otherwise nop
end
```

【０１０４】導入中にINSTALL COMMAND SCRIPT GENERATIONルーチンが生成したファイルを使用する方法の例を以下に示す。NWICHECKは、ターゲット・ワークステーション上での計画実行の順序を制御するプログラムである。

【０１０５】まず、NWICHECKが実行されると、このプログラムはワークステーション用の待ち行列ファイルを検査する。現行計画は待ち行列内の最初の計画になる。この計画はこの時点では待ち行列から除去されない。これは、ワークステーション上で計画が完了した後で行われる。この点からのＰＲＥ、ＰＯＳＴ、ＬＣＵの各コマンド・ファイルへの参照はすべて現行計画用になる。

【０１０６】ワークステーションがディスケットからブートされた場合、NWICHECKはそのワークステーション用のＬＣＵコマンド・ファイルを実行する。ワークステーションがハードファイルからブートされた場合、NWICHECKは、ハードファイル上にネットワーク導入ユーティリティを導入することになるＰＲＥコマンド・ファイルを実行する。システムは、リブートすると、自動的にＬＣＵコマンド・ファイルの実行を開始する。

【０１０７】ＬＣＵコマンド・ファイルは、プログラミングされた導入システムの実行を完了すると、SUCCESSフラグまたはFAILフラグを指定したＣｈｅｃｋＩｔ手続きを呼び出す。次に、ＣｈｅｃｋＩｔ手続きは、現行の計画名とSUCCESSまたはFAILフラグを指定してもう一度NWICHECKを呼び出す。NWICHECKは、計画名を指定して呼び出されると、まず、指定の計画用のＰＯＳＴコマンド・ファイルを実行する。FAILフラグを指定してNWICHECKが呼び出された場合は、処理は停止し、現行計画はそのまま待ち行列に残る。これは、訂正アクションを講じて、ワークステーション上で計画を再始動できるように行われる。成功フラグを指定してNWICHECKが呼び出された場合は、NWICHECKは指定の計画を待ち行列から除去する。待ち行列内に追加の要素がそれ以上ない場合、処理は停止し、追加の要素がまだある場合は、NWICHECKは待ち行列内の第１の要素に関するＰＲＥコマンド・ファイルを実行し、システムをリブートする。待ち行列内に追加の要素がそれ以上存在しなくなるか、または計画が失敗に終わるまで、処理は前述のように続行する。

【０１０８】以下の例について検討する。Ｗｋ１とＷｋ２という２つのワークステーションと、Plan1、Plan2、Plan3という３つの計画があるとする。Plan1では、オペレーティング・システムと、トランスポートと、リダイレクタを導入する。Plan2ではアプリケーションを導入し、Plan3ではアプリケーション３を導入する。ワークステーションＷｋ１では、Plan1とPlan2が実行される。ワークステーションＷｋ２では、Plan2とPlan3が実行される。ワークステーションＷｋ１はブート・ディスケットから始動され、ワークステーションＷｋ２はハードファイルから始動される。

【０１０９】ワークステーション１と２のワークステーション待ち行列ファイル：

Ｗｋ１
Plan1
Plan2
Ｗｋ２
Plan2
Plan3

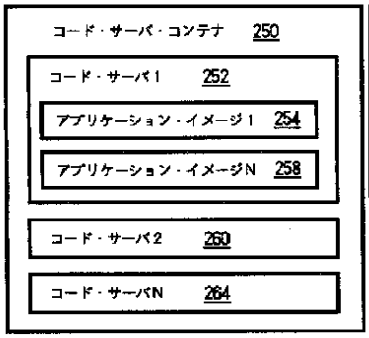
【０１１０】Plan1、Plan2、Plan3に含まれるアプリケーション：
Plan1
オペレーティング・システムを導入する
トランスポートを導入する
リダイレクタを導入する
Plan2
アプリケーション１を導入する
Plan2
アプリケーション２を導入する
【０１１１】ワークステーションＷｋ１用の処理：NWICHECKは、wk1の待ち行列を読み取り、Plan1を実行すべきであると判定する。次に、ワークステーションがディスケットからブートされたと判定し、Plan1用のwk1 LCU COMMAND FILEを実行する。wk1 LCU COMMAND FILEがオペレーティング・システムと、トランスポートと、リダイレクタとを導入した後、パラメータとしてPlan1とSUCCESSを指定してNWICHECKが再呼出しされる。NWICHECKは、Plan1用のwk1 POST COMMAND FILEを実行し、次にwk1待ち行列からPlan1を除去する。次にNWICHECKは、Plan2を実行すべきであると判定し、Plan2用のwk1 PRE COMMAND FILEを実行し、システムをリブートする。wk1 LCU COMMAND FILEがアプリケーション１を導入した後、パラメータとしてPlan2とSUCCESSを指定してNWICHECKが再呼出しされる。NWICHECKは、Plan2用のwk1 POST COMMAND FILEを実行し、wk1待ち行列からPlan2を除去する。次にNWICHECKは、wk1待ち行列内に追加の計画がそれ以上存在しないと判定し、処理が停止する。
【０１１２】ワークステーションＷｋ２用の処理：NWICHECKは、wk2の待ち行列ファイルを読み取り、Plan2を実行すべきであると判定する。次に、ワークステーションがハードファイルからブートされたと判定し、wk2 PRE COMMAND FILEを実行し、システムをリブートする。wk2 LCU COMMAND FILEがアプリケーション１を導入した後、パラメータとしてPlan2とSUCCESSを指定してNWICHECKが再呼出しされる。NWICHECKは、Plan2用のwk2 POST COMMANDFILEを実行し、次にwk2待ち行列からPlan2を除去する。次にNWICHECKは、Plan3を実行すべきであると判定し、Plan3用のwk2 PRE COMMAND FILEを実行し、システムをリブートする。wk2 LCU COMMAND FILEがアプリケーション２を導入した後、NWICHECKは、Plan3用のwk2 POST COMMAND FILEを実行し、wk2待ち行列からPlan3を除去する。次にNWICHECKは、wk2待ち行列内に追加の計画がそれ以上存在しないと判定し、処理が停止する。
【０１１３】まとめとして、本発明の構成に関して以下の事項を開示する。
【０１１４】（１）ネットワークでアプリケーションを導入するための導入計画オブジェクトをコミットする方法において、導入計画オブジェクト内の子オブジェクトを検査し、必要であれば、導入計画オブジェクトに追加の子オブジェクトを追加することにより、導入計画オブジェクトを事前妥当性検査するステップと、導入計画オブジェクトとその子オブジェクト内のデータを検査して、データにエラーがないかどうか確認することにより、導入計画オブジェクトを妥当性検査するステップと、導入計画が正常に妥当性検査された場合に、導入計画オブジェクト内の子オブジェクトを、データ構造に応じてネットワークでアプリケーションを導入するネットワーク導入エンジンに使用可能なデータ構造に変換するステップとを含む方法。
（２）導入計画オブジェクトが、アプリケーション・プログラムを表す計画中アプリケーション・オブジェクトと、アプリケーション・プログラムを導入すべきワークステーションのグループを表す計画中グループ・オブジェクトとを含むことを特徴とする、上記（１）に記載の方法。
（３）導入計画が、アプリケーションの導入用のデータを含む応答ファイル・オブジェクトと、特定のワークステーション用の応答ファイル・オブジェクト・データをカストマイズするためのデータを含むカストマイズ・ファイル・オブジェクトとをさらに含むことを特徴とする、上記（２）に記載の方法。
（４）妥当性検査ステップが、導入計画オブジェクト内のデータとして指定されたターゲット・ワークステーション上のファイル・ディレクトリを検査して、そのファイル・ディレクトリが物理的に有効でネットワーク上でアクセス可能であるかどうかを判定することをさらに含み、ターゲット・ワークステーションが、そのアプリケーションを導入すべきワークステーションのグループのうちの１つであることを特徴とする、上記（２）に記載の方法。
（５）妥当性検査ステップが、ワークステーションのグループ上でのアプリケーションの導入に必要なコード・サーバ・ワークステーション上のオブジェクトの存在を確認することをさらに含み、コード・サーバ・オブジェクトが、導入計画オブジェクトの子オブジェクトではないことを特徴とする、上記（２）に記載の方法。
（６）妥当性検査ステップが、アプリケーション・オブジェクトが、ネットワーク内の第１のアプリケーションに更新内容を提供するアプリケーションを表していることを判定するステップと、アプリケーション・コンテナ・オブジェクト内の第１のアプリケーションを表す第１のアプリケーション・オブジェクトの存在を確認するス

テップとをさらに含むことを特徴とする、上記（２）に記載の方法。
（７）妥当性検査ステップが、ターゲット・ワークステーション上のオペレーティング・システムとその訂正サービスのレベルとを確認することをさらに含むことを特徴とする、上記（２）に記載の方法。
（８）導入計画オブジェクトが、導入すべき複数のアプリケーションと対応する応答ファイルとをそれぞれ表す、複数の計画中アプリケーション・オブジェクトと複数の対応する応答ファイル・オブジェクトとを含み、変換ステップが、ワークステーションのグループ内の特定のワークステーションに応じて複数の応答ファイル・オブジェクトをカストマイズするために、カストマイズ・ファイル・オブジェクトを複数の応答ファイル・オブジェクトに適用することをさらに含むことを特徴とする、上記（３）に記載の方法。
（９）複数の導入計画オブジェクトについて、事前妥当性検査ステップと、妥当性検査ステップと、変換ステップとを繰り返すステップと、各ワークステーションごとにワークステーション待ち行列に複数の導入計画オブジェクトを待ち行列化するステップとをさらに含むことを特徴とする、上記（１）に記載の方法。
（１０）ネットワークでアプリケーションを導入するための導入計画オブジェクトをコミットするシステムにおいて、導入計画オブジェクト内の子オブジェクトを検査し、必要であれば、導入計画オブジェクトに追加の子オブジェクトを追加することにより、導入計画オブジェクトを事前妥当性検査する手段と、導入計画オブジェクトとその子オブジェクト内のデータを検査して、データにエラーがないかどうか確認することにより、導入計画オブジェクトを妥当性検査する手段と、導入計画が正常に妥当性検査された場合に、導入計画オブジェクト内の子オブジェクトを、データ構造に応じてネットワークでアプリケーションを導入するネットワーク導入エンジンに使用可能なデータ構造に変換する手段とを含むシステム。
（１１）導入計画オブジェクトが、アプリケーション・プログラムを表す計画中アプリケーション・オブジェクトと、アプリケーション・プログラムを導入すべきワークステーションのグループを表す計画中グループ・オブジェクトとを含むことを特徴とする、上記（１０）に記載のシステム。
（１２）導入計画が、アプリケーションの導入用のデータを含む応答ファイル・オブジェクトと、特定のワークステーション用の応答ファイル・オブジェクト・データをカストマイズするためのデータを含むカストマイズ・ファイル・オブジェクトとをさらに含むことを特徴とする、上記（１１）に記載のシステム。
（１３）妥当性検査手段が、導入計画オブジェクト内のデータとして指定されたターゲット・ワークステーション上のファイル・ディレクトリを検査して、そのファイル・ディレクトリが物理的に有効でネットワーク上でアクセス可能であるかどうかを判定する手段をさらに含み、ターゲット・ワークステーションが、そのアプリケーションを導入すべきワークステーションのグループのうちの１つであることを特徴とする、上記（１１）に記載のシステム。
（１４）妥当性検査手段が、ワークステーションのグループ上でのアプリケーションの導入に必要なコード・サーバ・ワークステーション上のオブジェクトの存在を確認する手段をさらに含み、コード・サーバ・オブジェクトが、導入計画オブジェクトの子オブジェクトではないことを特徴とする、上記（１１）に記載のシステム。
（１５）妥当性検査手段が、アプリケーション・オブジェクトが、ネットワーク内の第１のアプリケーションに更新内容を提供するアプリケーションを表していることを判定する手段と、アプリケーション・コンテナ・オブジェクト内の第１のアプリケーションを表す第１のアプリケーション・オブジェクトの存在を確認する手段とをさらに含むことを特徴とする、上記（１１）に記載のシステム。
（１６）導入計画オブジェクトが、導入すべき複数のアプリケーションと対応する応答ファイルとをそれぞれ表す、複数の計画中アプリケーション・オブジェクトと複数の対応する応答ファイル・オブジェクトとを含み、変換ステップが、ワークステーションのグループ内の特定のワークステーションに応じて複数の応答ファイル・オブジェクトをカストマイズするために、カストマイズ・ファイル・オブジェクトを複数の応答ファイル・オブジェクトに適用することをさらに含むことを特徴とする、上記（１２）に記載のシステム。
（１７）複数の導入計画オブジェクトについて、事前妥当性検査ステップと、妥当性検査ステップと、変換ステップとを繰り返す手段と、各ワークステーションごとにワークステーション待ち行列に複数の導入計画オブジェクトを待ち行列化する手段とをさらに含むことを特徴とする、上記（１０）に記載のシステム。
（１８）ネットワークでアプリケーションを導入するための導入計画オブジェクトをコミットするためにコンピュータで読取り可能な命令を格納するコンピュータ・メモリにおいて、導入計画オブジェクト内の子オブジェクトを検査し、必要であれば、導入計画オブジェクトに追加の子オブジェクトを追加することにより、導入計画オブジェクトを事前妥当性検査する手段と、導入計画オブジェクトとその子オブジェクト内のデータを検査して、データにエラーがないかどうか確認することにより、導入計画オブジェクトを妥当性検査する手段と、導入計画が正常に妥当性検査された場合に、導入計画オブジェクト内の子オブジェクトを、データ構造に応じてネットワークでアプリケーションを導入するネットワーク導入エ

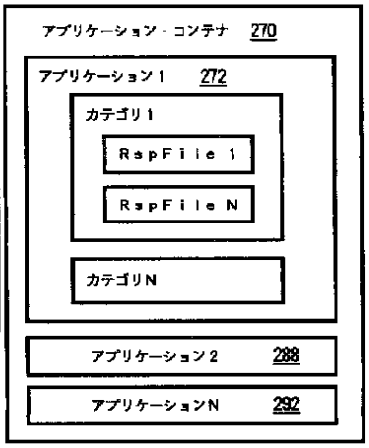
ンジンに使用可能なデータ構造に変換する手段とを含み、コンピュータ・メモリがコンピュータ・システムに結合され、コンピュータ・システムによってアクセスされたときに、その手段が活動化されることを特徴とするコンピュータ・メモリ。
（１９）導入計画オブジェクトが、アプリケーション・プログラムを表す計画中アプリケーション・オブジェクトと、アプリケーション・プログラムを導入すべきワークステーションのグループを表す計画中グループ・オブジェクトとを含むことを特徴とする、上記（１８）に記載のコンピュータ・メモリ。
（２０）導入計画が、アプリケーションの導入用のデータを含む応答ファイル・オブジェクトと、特定のワークステーション用の応答ファイル・オブジェクト・データをカストマイズするためのデータを含むカストマイズ・ファイル・オブジェクトとをさらに含むことを特徴とする、上記（１９）に記載のコンピュータ・メモリ。
（２１）妥当性検査手段が、導入計画オブジェクト内のデータとして指定されたターゲット・ワークステーション上のファイル・ディレクトリを検査して、そのファイル・ディレクトリが物理的に有効でネットワーク上でアクセス可能であるかどうかを判定する手段をさらに含み、ターゲット・ワークステーションが、そのアプリケーションを導入すべきワークステーションのグループのうちの１つであることを特徴とする、上記（１９）に記載のコンピュータ・メモリ。
（２２）妥当性検査手段が、ワークステーションのグループ上でのアプリケーションの導入に必要なコード・サーバ・ワークステーション上のオブジェクトの存在を確認する手段をさらに含み、コード・サーバ・オブジェクトが、導入計画オブジェクトの子オブジェクトではないことを特徴とする、上記（１９）に記載のコンピュータ・メモリ。
（２３）妥当性検査手段が、アプリケーション・オブジェクトが、ネットワーク内の第１のアプリケーションに更新内容を提供するアプリケーションを表していることを判定する手段と、アプリケーション・コンテナ・オブジェクト内の第１のアプリケーションを表す第１のアプリケーション・オブジェクトの存在を確認する手段とをさらに含むことを特徴とする、上記（１９）に記載のコンピュータ・メモリ。
（２４）導入計画オブジェクトが、導入すべき複数のアプリケーションと対応する応答ファイルとをそれぞれ表す、複数の計画中アプリケーション・オブジェクトと複数の対応する応答ファイル・オブジェクトとを含み、変換ステップが、ワークステーションのグループ内の特定のワークステーションに応じて複数の応答ファイル・オブジェクトをカストマイズするために、カストマイズ・ファイル・オブジェクトを複数の応答ファイル・オブジェクトに適用することをさらに含むことを特徴とする、上記（２０）に記載のコンピュータ・メモリ。
（２５）複数の導入計画オブジェクトについて、事前妥当性検査ステップと、妥当性検査ステップと、変換ステップとを繰り返す手段と、各ワークステーションごとにワークステーション待ち行列に複数の導入計画オブジェクトを待ち行列化する手段とをさらに含むことを特徴とする、上記（１８）に記載のコンピュータ・メモリ。
【図面の簡単な説明】
【図１】本発明の教示により構成したコンピュータ・システムを示す図である。
【図２】導入計画オブジェクトの階層表現を示す図である。
【図３】導入計画オブジェクト内のオブジェクトに関連するオブジェクトを示す図である。
【図４】導入計画オブジェクト内のオブジェクトに関連するオブジェクトを示す図である。
【図５】導入計画オブジェクト内のオブジェクトに関連するオブジェクトを示す図である。
【図６】導入計画オブジェクト内のオブジェクトに関連するオブジェクトを示す図である。
【図７】コミット・プロセスの流れ図である。
【図８】事前妥当性検査プロセスの流れ図である。
【図９】導入計画オブジェクトに関する妥当性検査プロセスの流れ図である。
【図１０】導入計画オブジェクト内の応答ファイルにカストマイズ・ファイル・オブジェクトを適用するプロセスを示す図である。
【図１１】計画中アプリケーション・オブジェクトに関する妥当性検査プロセスを示す図である。
【図１２】計画中アプリケーション・オブジェクトに保守システムが必要かどうかを判定するためのプロセスの流れ図である。
【図１３】計画に含まれないが計画内のオブジェクトに関連するアプリケーション・オブジェクトに関する妥当性検査プロセスの流れ図である。
【図１４】計画中ワークステーション・オブジェクトに関する妥当性検査プロセスを示す図である。
【図１５】計画中ワークステーション・アプリケーション・オブジェクトに関する妥当性検査プロセスを示す図である。
【図１６】計画中カテゴリ・オブジェクトに関する妥当性検査プロセスの流れ図である。
【図１７】応答ファイルとカストマイズ・ファイル・オブジェクトの妥当性検査を示す図である。
【図１８】計画中グループ・オブジェクトの妥当性検査の流れ図である。
【図１９】コード・サーバ・オブジェクトに関する妥当性検査プロセスを示す図である。
【図２０】コード・サーバ・オブジェクト用の計画中アプリケーション・オブジェクトについてアプリケーショ

ン・イメージが存在するかどうかを検査するためのプロセスを示す図である。
【図２１】計画出力ファイルを示す図である。
【図２２】計画状況ファイルを示す図である。
【図２３】ワークステーション待ち行列ファイルを示す図である。
【図２４】ＰＲＥコマンド・ファイルを示す図である。
【図２５】ＰＯＳＴコマンド・ファイルを示す図である。
【図２６】ＬＣＵコマンド・ファイルを示す図である。
【図２７】導入コマンド・スクリプトを生成するためのプロセスの流れ図である。
【図２８】ＬＣＵコマンド作成プロセスの流れ図である。
【図２９】可変セクションを書き込むためのプロセスを示す図である。
【図３０】製品データ・セクションを書き込むためのプロセスを示す図である。
【図３１】書込み検査手続きを示す図である。
【図３２】導入セクションを書き込むための流れ図である。
【図３３】アプリケーション・ルーチンを書き込むための流れ図である。
【図３４】導入実行書込みルーチンの流れ図である。
【図３５】最終アプリケーション開始ルーチンの流れ図である。
【図３６】終了セクション開始ルーチンの流れ図である。
【図３７】保守セクション・プロセスを示す図である。
【図３８】ブート・ディスク・プロセスを示す図である。
【図３９】ハード・ディスク・プロセスを示す図である。
【符号の説明】
６００　導入計画用のVALIDATEを開始する
６０５　計画が少なくとも１つのＡｐｐＩＰを含むかどうか
６１０　Ｉ＝１
６１５　Ｉ＜＝ＡｐｐＩＰの数であるかどうか
６２０　VALIDATEメソッドをＡｐｐＩＰ．Ｉに送る
６２５　Ｉ＝Ｉ＋１
６３０　VALERROR=VALERROR+1
６３５　計画がＧｒｏｕｐＩＰを含むかどうか
６４０　VALIDATEメソッドをＧｒｏｕｐＩＰを送る
６４５　VALERROR=VALERROR+1
６５０　計画用のローカル・ディレクトリが存在するかどうか
６５５　VALERROR=VALERROR+1
６６０　Ｉ＝１
６６５　Ｉ＜＝コード・サーバ・オブジェクトの数であるかどうか
６７０　VALIDATEメソッドをCODESERVER.Iに送る
６７５　Ｉ＝Ｉ＋１
６８０　VALIDATEメソッドから復帰する

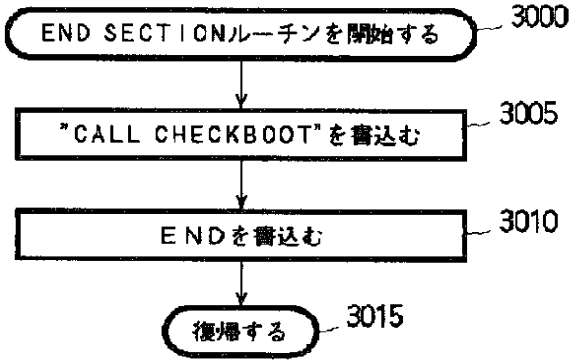
【図３】



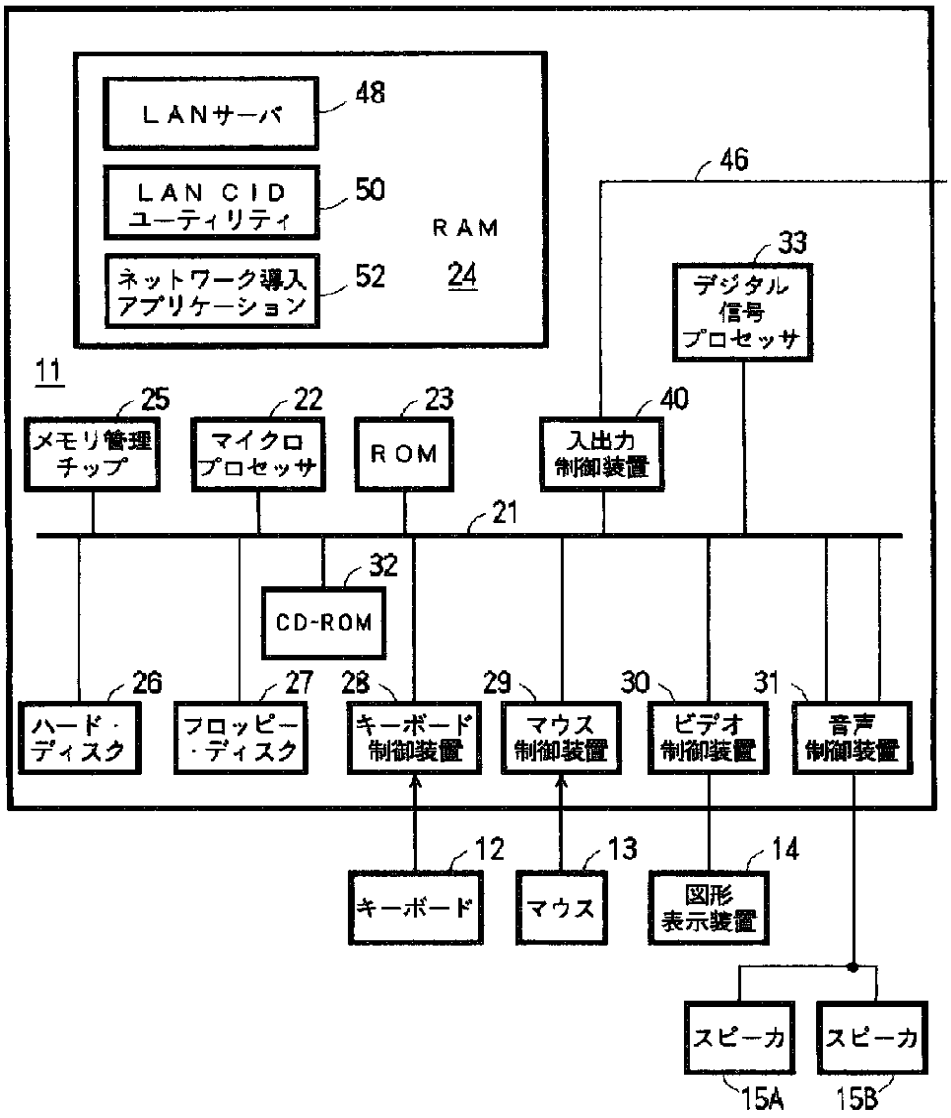
【図４】



【図３６】



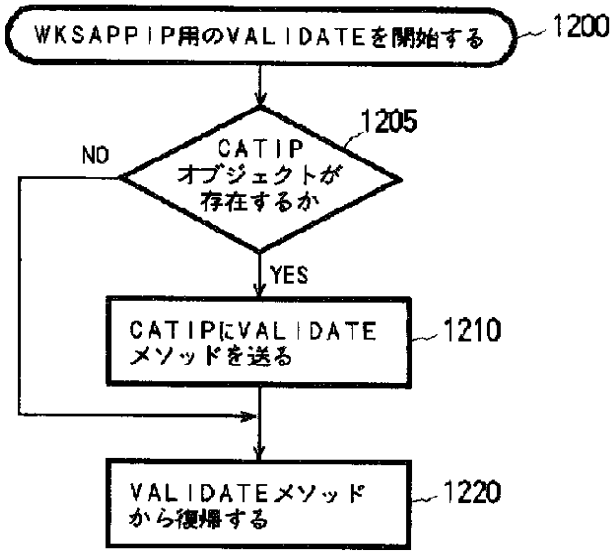
【図１】


LANサーバ
48
LAN CID
ユーティリティ
50
ネットワーク導入
アプリケーション
52
RAM
24
46
33
デジタル
信号
プロセッサ
11
25
メモリ管理
チップ
22
マイクロ
プロセッサ
23
ROM
40
入出力
制御装置
21
32
CD-ROM
26
ハード・
ディスク
27
フロッピー
・ディスク
28
キーボード
制御装置
29
マウス
制御装置
30
ビデオ
制御装置
31
音声
制御装置
12
キーボード
13
マウス
14
図形
表示装置
スピーカ
スピーカ
15A
15B

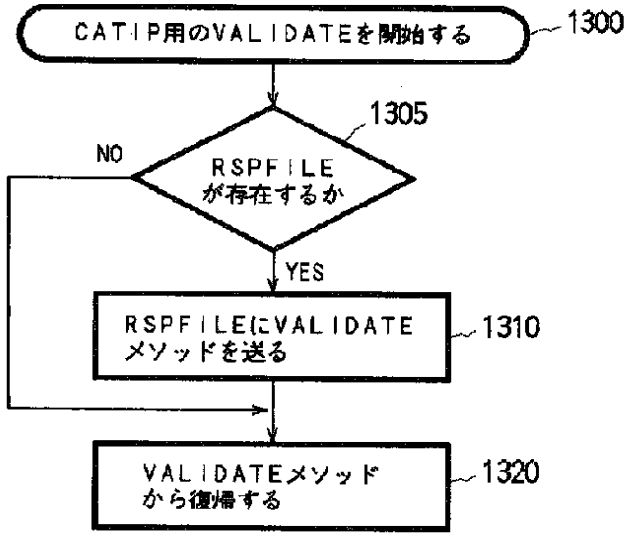
【図2】

計　画	200
APPIP 1	202
CUSTFILE 1	204
CATIP 1	206
RSPFILE 1	208
APPIP 2	210
⋮	212
APPIP N	214
CATIP N	216
RSPFILE N	218
GROUPIP	220
WKSIP 1	222
WKSAPPIP 1	224
CATIP 1	226
RSPFILE 1	228
WKSAPPIP 2	230
⋮	232
WKSAPPIP N	234
CATIP N	236
RSPFILE N	238
WKSIP 2	240
⋮	242
WKSIP M	244

【図15】



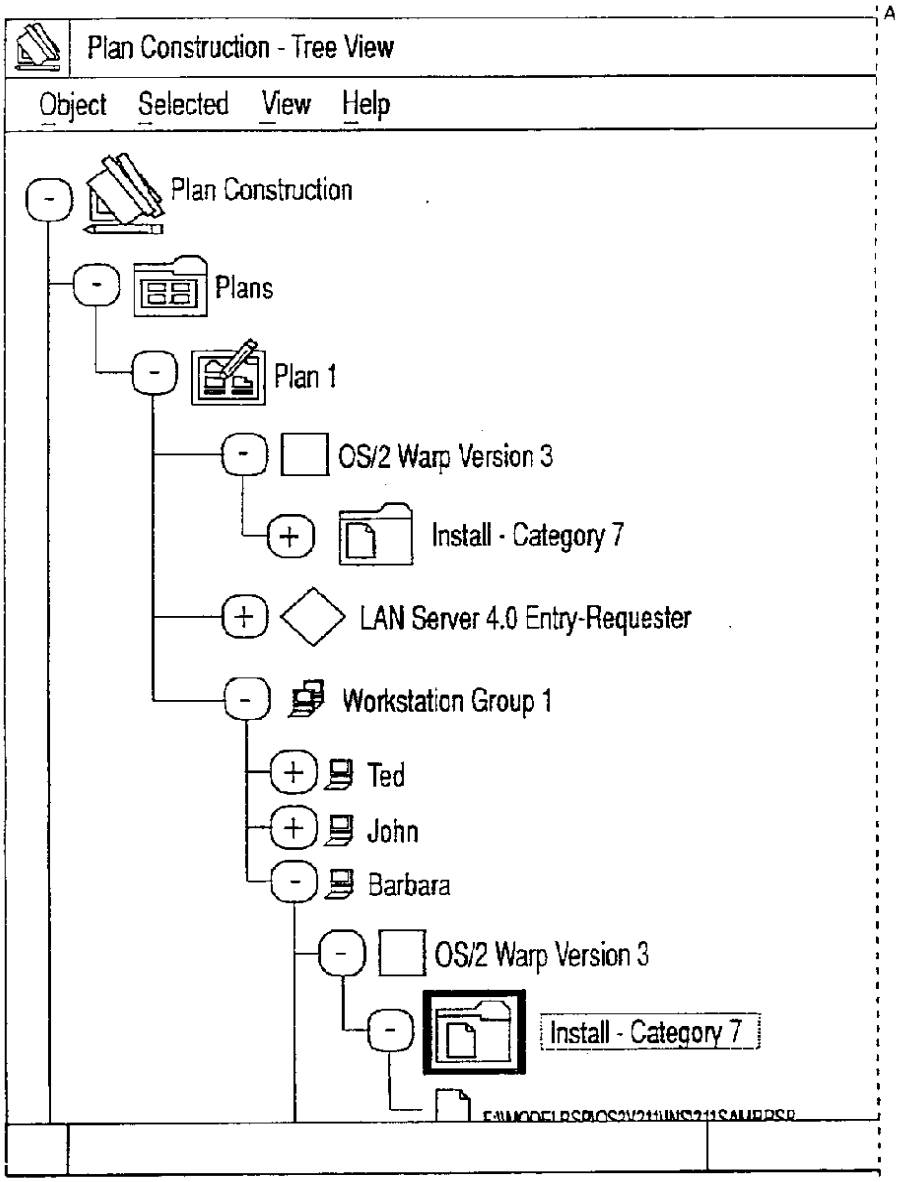
【図16】



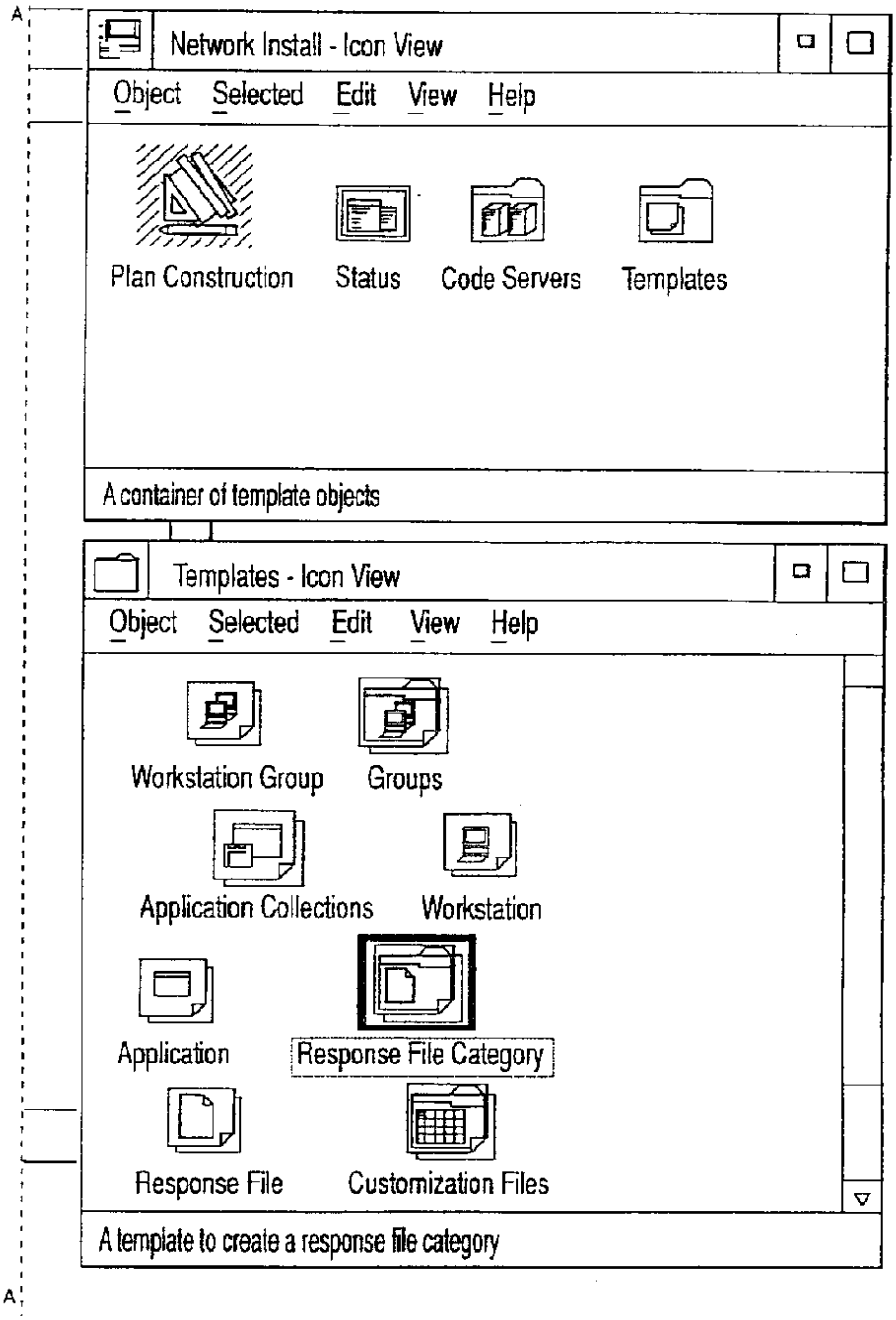
【図23】

ワークステーション待ち行列ファイル	2060
計画名 1	2062
・・・	2064
計画名 N	2066

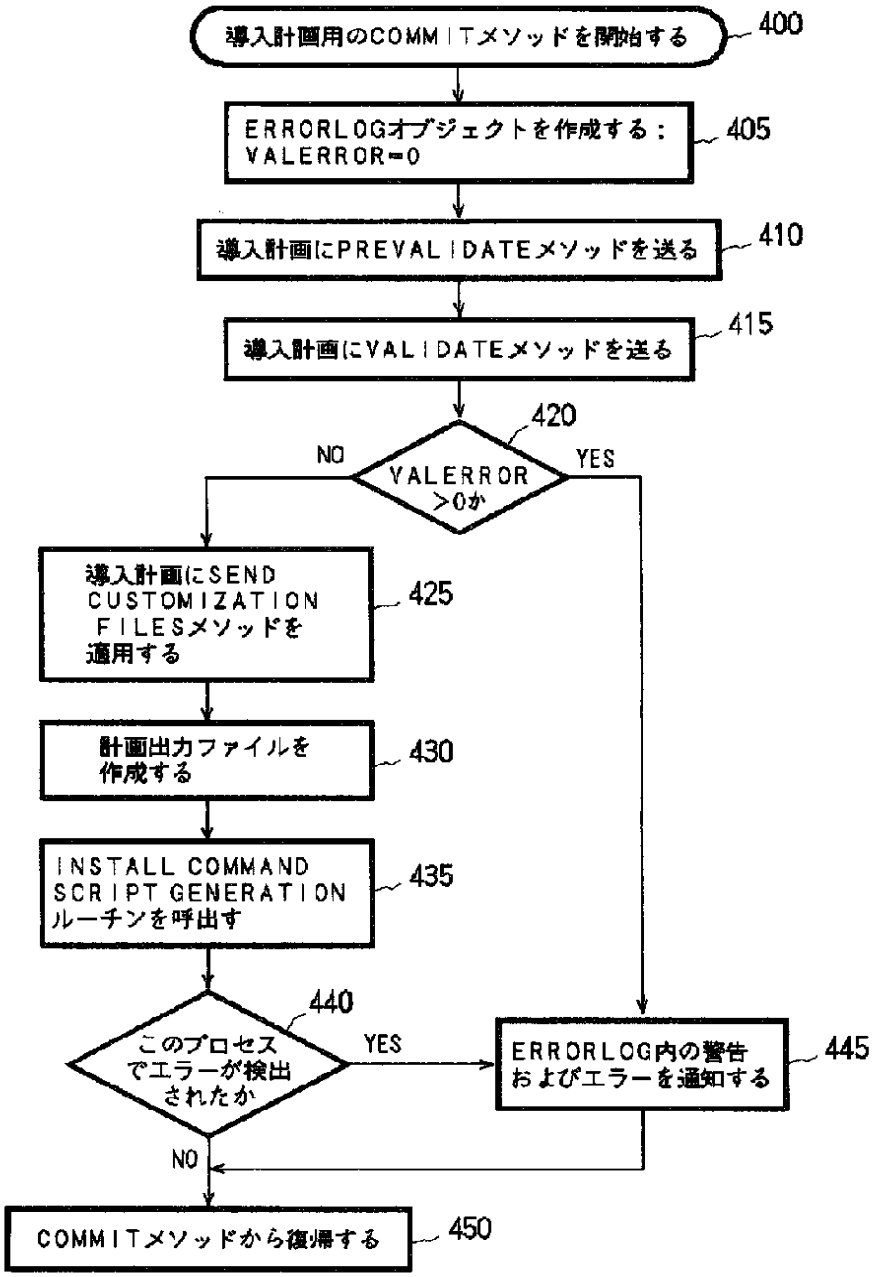
【図5】


Plan Construction - Tree View
Object Selected View Help
Plan Construction
Plans
Plan 1
OS/2 Warp Version 3
Install - Category 7
LAN Server 4.0 Entry-Requester
Workstation Group 1
Ted
John
Barbara
OS/2 Warp Version 3
Install - Category 7
計画中の応答ファイルのコンテナ

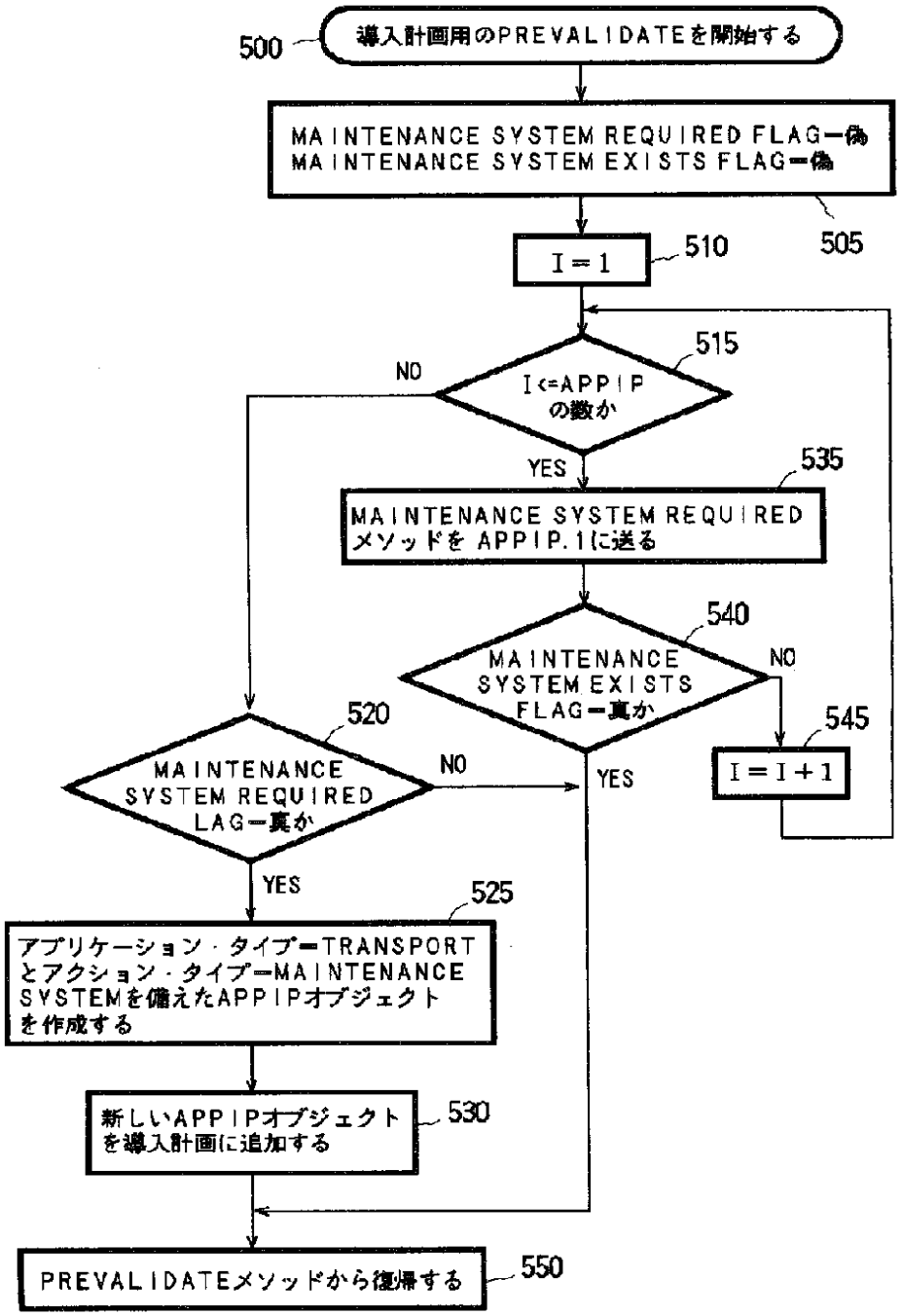
【図６】


A
Network Install - Icon View
Object Selected Edit View Help
Plan Construction
Status
Code Servers
Templates
A container of template objects
Templates - Icon View
Object Selected Edit View Help
Workstation Group
Groups
Application Collections
Workstation
Application
Response File Category
Response File
Customization Files
A template to create a response file category
A

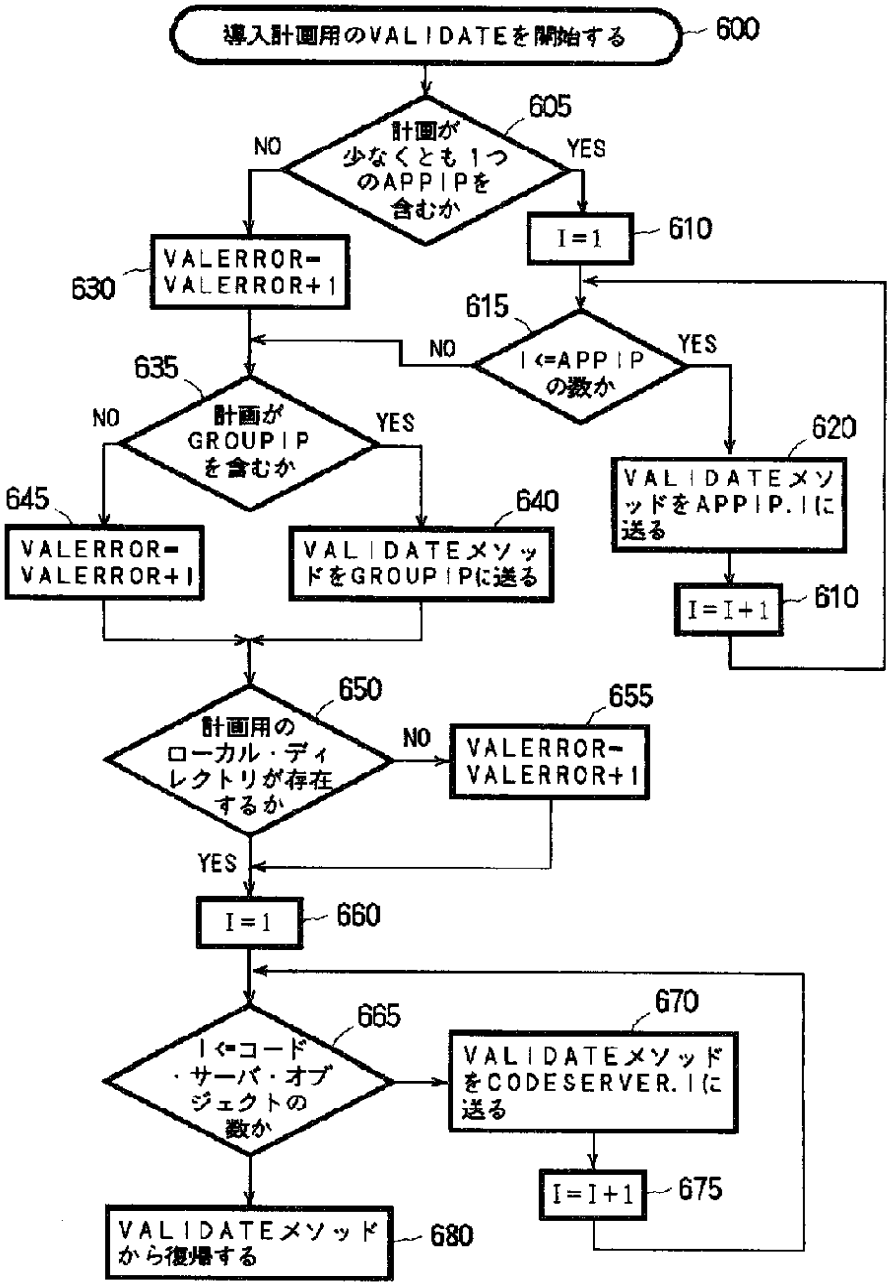
【図７】


導入計画用のCOMMITメソッドを開始する
400
ERRORLOGオブジェクトを作成する：VALERROR=0
405
導入計画にPREVALIDATEメソッドを送る
410
導入計画にVALIDATEメソッドを送る
415
420
NO
VALERROR >0か
YES
導入計画にSEND CUSTOMIZATION FILESメソッドを適用する
425
計画出力ファイルを作成する
430
INSTALL COMMAND SCRIPT GENERATIONルーチンを呼出す
435
440
このプロセスでエラーが検出されたか
YES
ERRORLOG内の警告およびエラーを通知する
445
NO
COMMITメソッドから復帰する
450

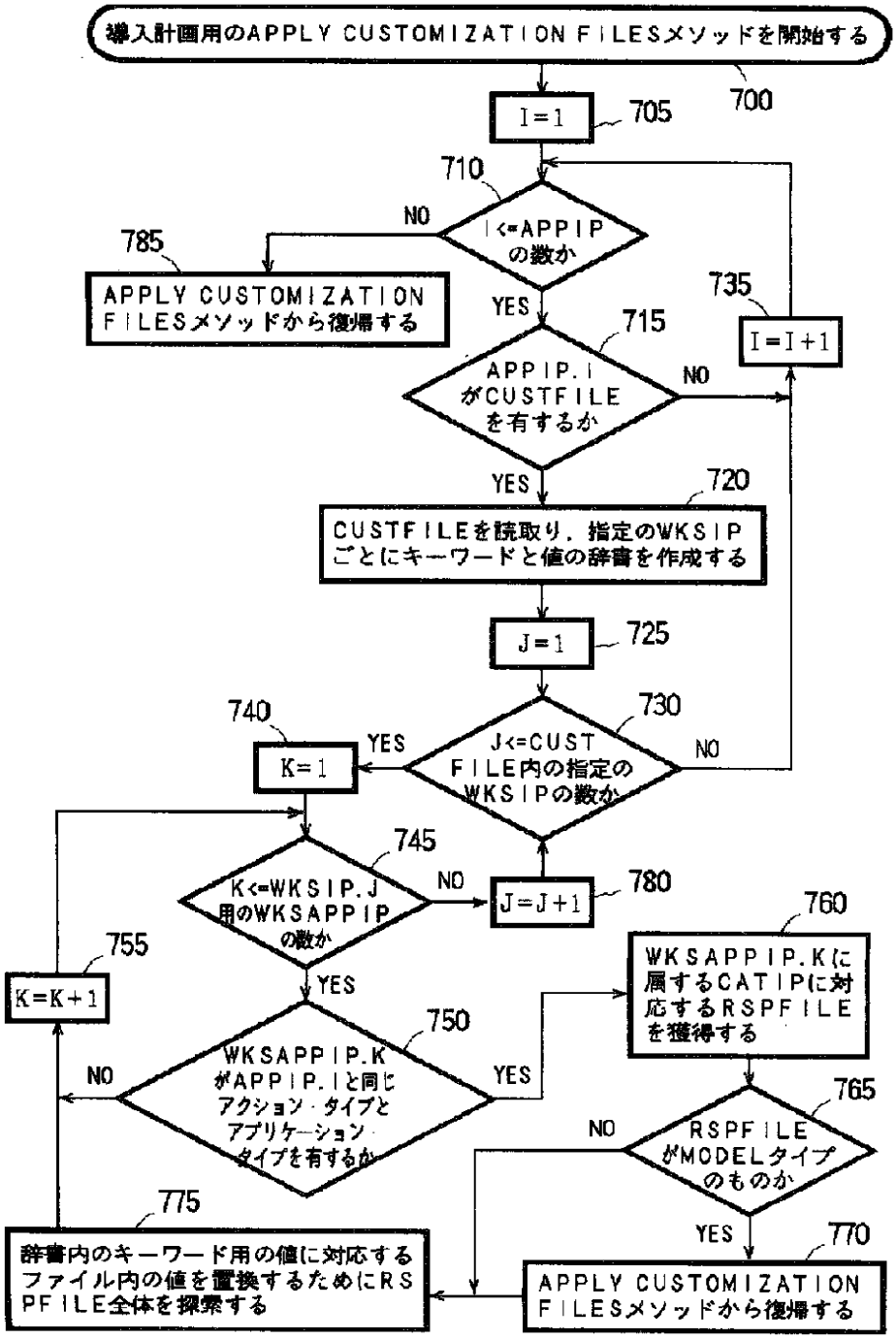
【図８】


500
導入計画用のPREVALIDATEを開始する
MAINTENANCE SYSTEM REQUIRED FLAG←偽
MAINTENANCE SYSTEM EXISTS FLAG←偽
505
I＝1
510
515
NO
I<=APPIPの数か
YES
535
MAINTENANCE SYSTEM REQUIRED メソッドを APPIP.1に送る
540
MAINTENANCE SYSTEM EXISTS FLAG←真か
NO
545
I＝I＋1
520
MAINTENANCE SYSTEM REQUIRED LAG←真か
NO
YES
YES
525
アプリケーション・タイプ←TRANSPORT とアクション・タイプ←MAINTENANCE SYSTEMを備えたAPPIPオブジェクト を作成する
新しいAPPIPオブジェクト を導入計画に追加する
530
PREVALIDATEメソッドから復帰する
550

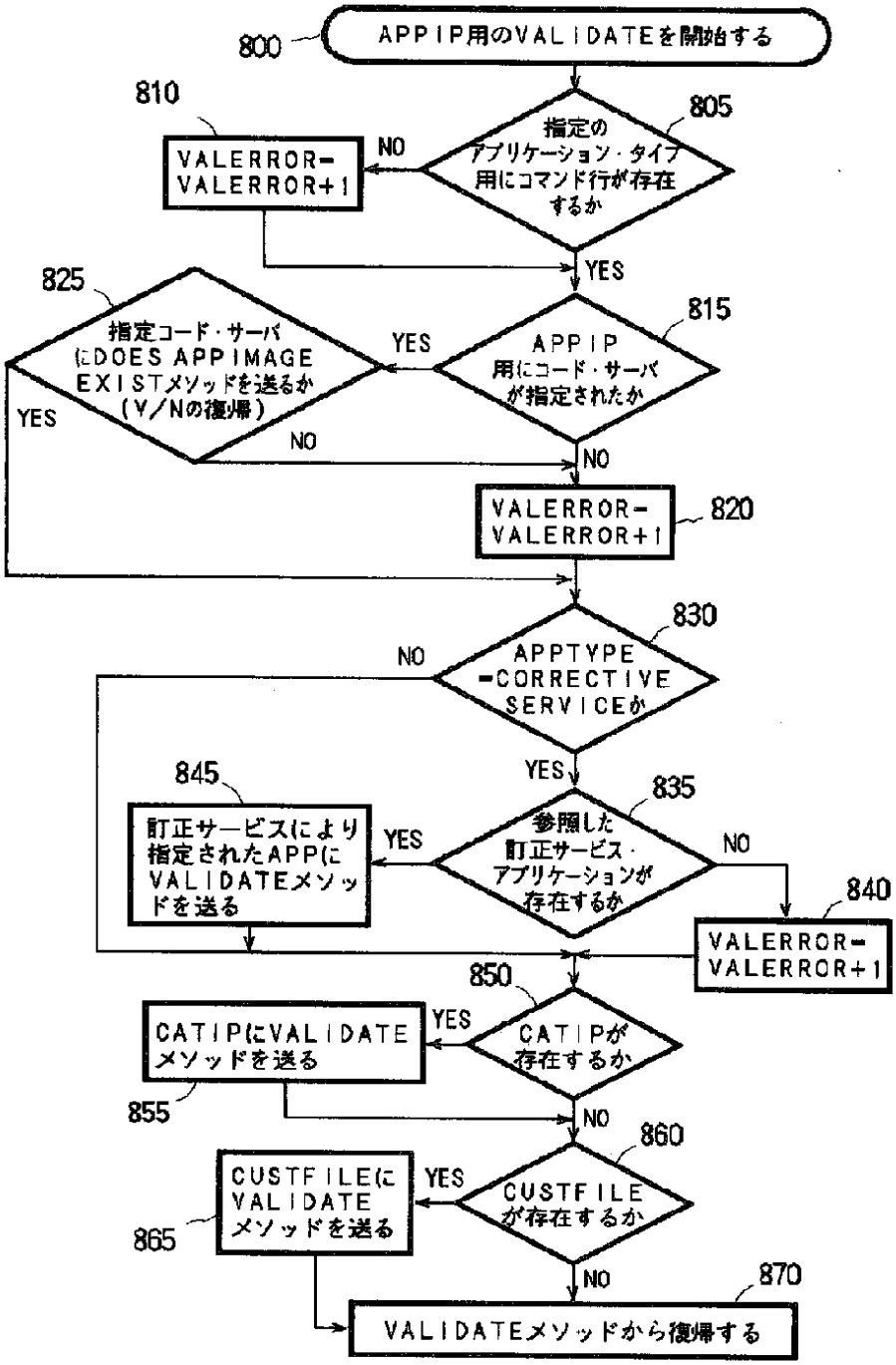
【図９】


導入計画用のVALIDATEを開始する
600
計画が少なくとも1つのAPPIPを含むか
605
NO
YES
I=1
610
VALERROR=VALERROR+1
630
615
I<=APPIPの数か
NO
YES
635
計画がGROUPIPを含むか
NO
YES
620
VALIDATEメソッドをAPPIP.Iに送る
645
VALERROR=VALERROR+1
640
VALIDATEメソッドをGROUPIPに送る
I=I+1
610
650
計画用のローカル・ディレクトリが存在するか
NO
655
VALERROR=VALERROR+1
YES
I=1
660
665
I<=コード・サーバ・オブジェクトの数か
670
VALIDATEメソッドをCODESERVER.Iに送る
I=I+1
675
VALIDATEメソッドから復帰する
680

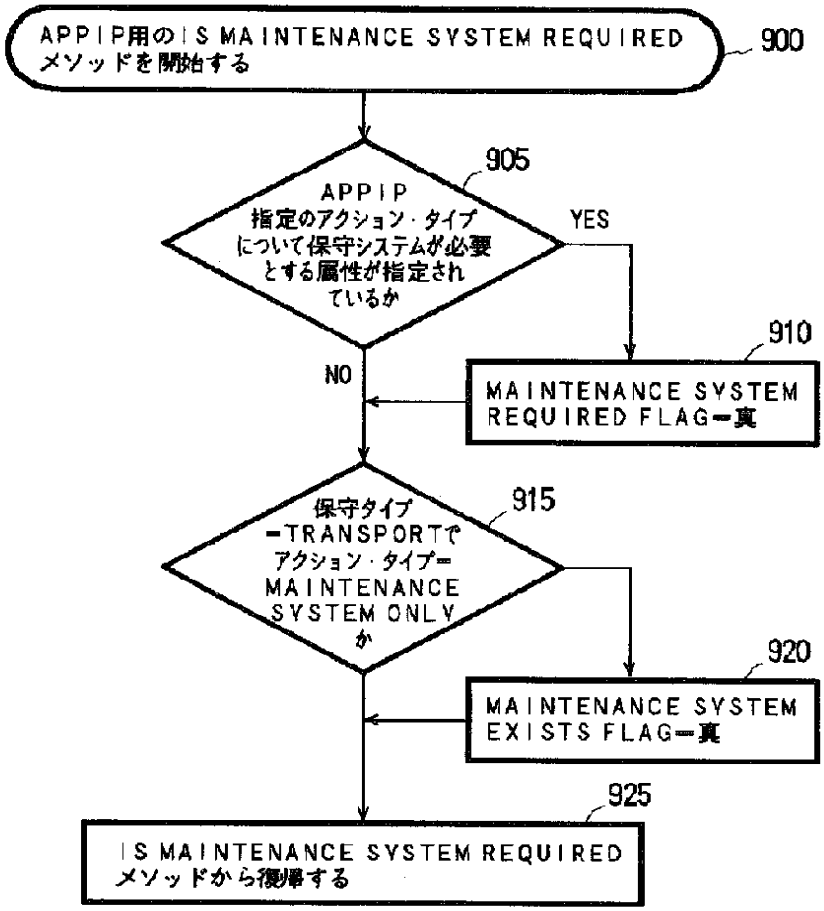
【図１０】


導入計画用のAPPLY CUSTOMIZATION FILESメソッドを開始する
700
I=1
705
710
I<=APPIPの数か
NO
785
APPLY CUSTOMIZATION FILESメソッドから復帰する
YES
715
APPIP.IがCUSTFILEを有するか
NO
735
I=I+1
YES
720
CUSTFILEを読取り，指定のWKSIPごとにキーワードと値の辞書を作成する
J=1
725
740
K=1
YES
730
J<=CUST FILE内の指定のWKSIPの数か
NO
745
K<=WKSIP.J用のWKSAPPIPの数か
NO
780
J=J+1
760
WKSAPPIP.Kに属するCATIPに対応するRSPFILEを獲得する
755
K=K+1
YES
750
WKSAPPIP.KがAPPIP.Iと同じアクション・タイプとアプリケーション・タイプを有するか
NO
YES
765
RSPFILEがMODELタイプのものか
NO
YES
775
辞書内のキーワード用の値に対応するファイル内の値を置換するためにRSPFILE全体を探索する
770
APPLY CUSTOMIZATION FILESメソッドから復帰する

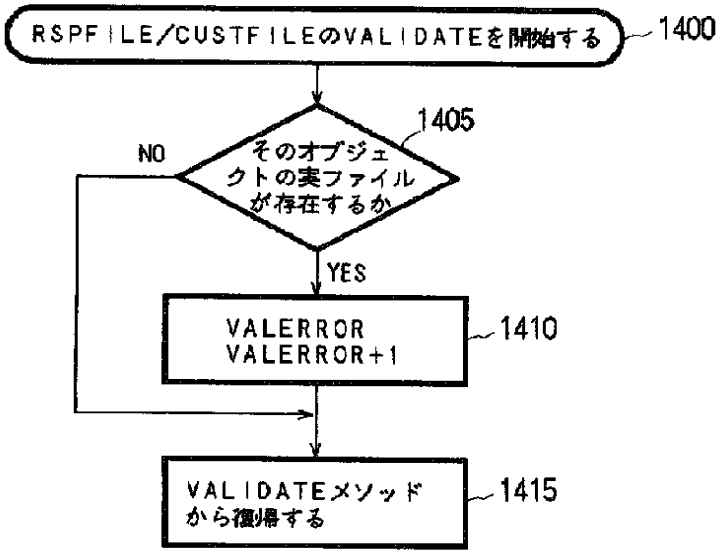
【図１１】


800
APPIP用のVALIDATEを開始する
805
指定のアプリケーション・タイプ用にコマンド行が存在するか
NO
810
VALERROR=VALERROR+1
YES
815
APPIP用にコード・サーバが指定されたか
YES
825
指定コード・サーバにDOES APPIMAGE EXISTメソッドを送るか（Y/Nの復帰）
YES
NO
NO
820
VALERROR=VALERROR+1
830
APPTYPE=CORRECTIVE SERVICEか
NO
YES
835
参照した訂正サービス・アプリケーションが存在するか
YES
845
訂正サービスにより指定されたAPPにVALIDATEメソッドを送る
NO
840
VALERROR=VALERROR+1
850
CATIPが存在するか
YES
855
CATIPにVALIDATEメソッドを送る
NO
860
CUSTFILEが存在するか
YES
865
CUSTFILEにVALIDATEメソッドを送る
NO
870
VALIDATEメソッドから復帰する

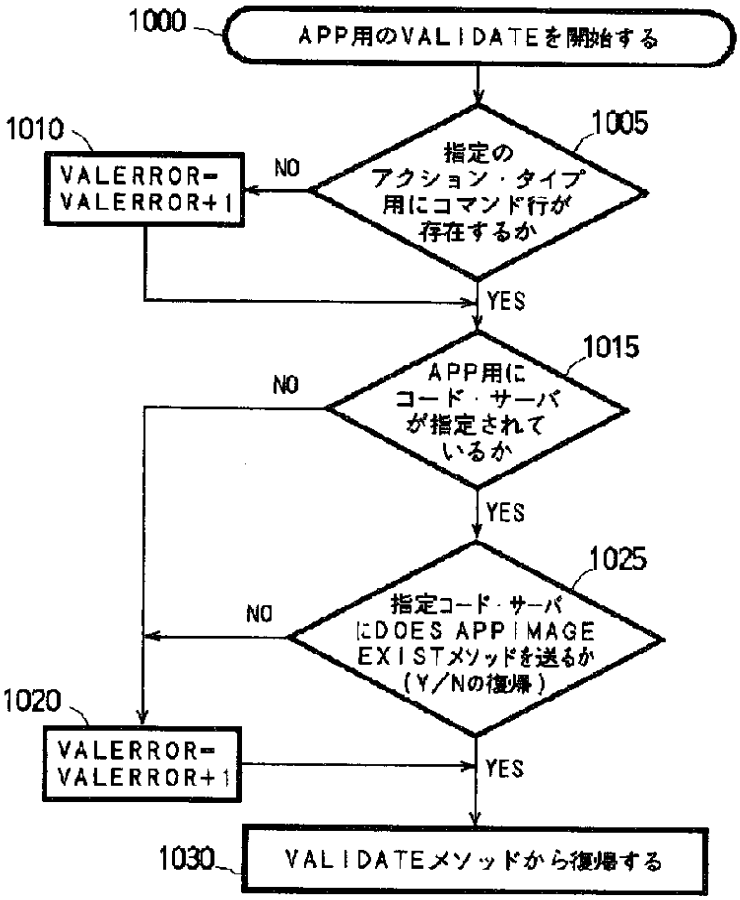
【図１２】


APPIP用のIS MAINTENANCE SYSTEM REQUIRED メソッドを開始する
900
APPIP 指定のアクション・タイプについて保守システムが必要とする属性が指定されているか
905
YES
NO
MAINTENANCE SYSTEM REQUIRED FLAG←真
910
保守タイプ←TRANSPORTでアクション・タイプ=MAINTENANCE SYSTEM ONLYか
915
MAINTENANCE SYSTEM EXISTS FLAG←真
920
IS MAINTENANCE SYSTEM REQUIRED メソッドから復帰する
925

【図１７】


RSPFILE/CUSTFILEのVALIDATEを開始する
1400
そのオブジェクトの実ファイルが存在するか
1405
NO
YES
VALERROR VALERROR+1
1410
VALIDATEメソッドから復帰する
1415

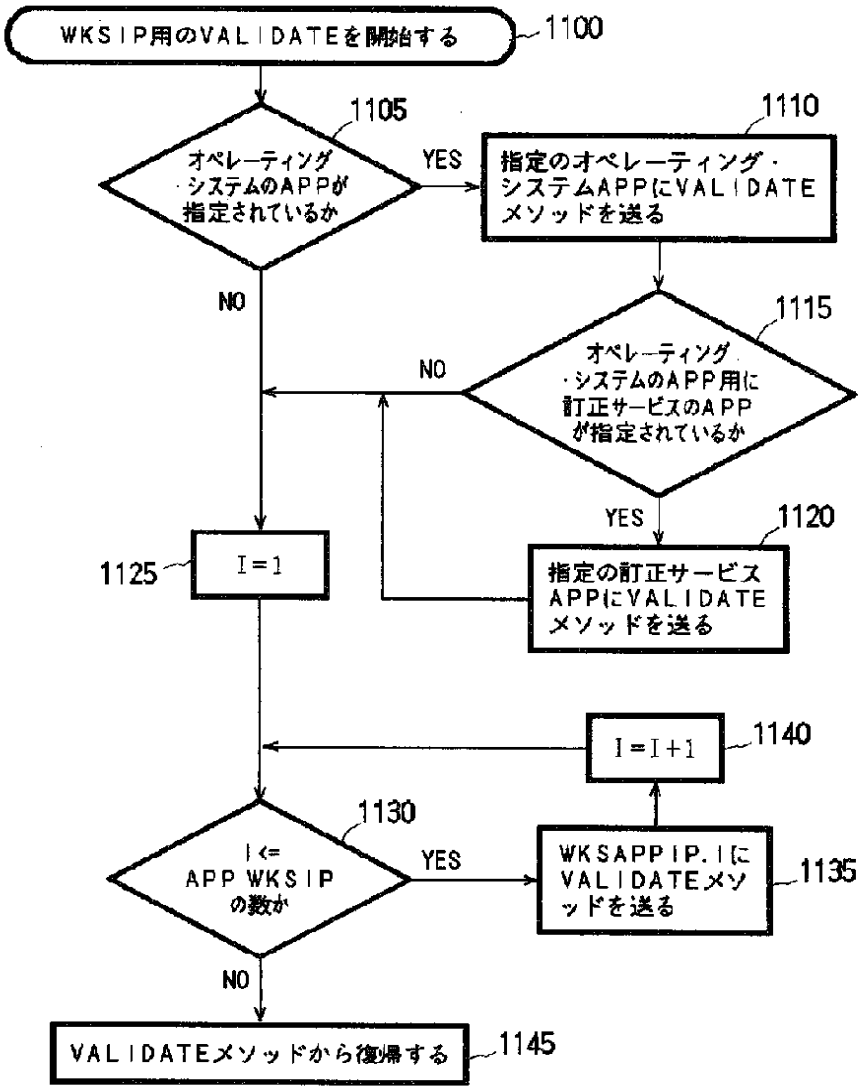
【図１３】



【図２２】

計画状況ファイル	2030
計画セクション	2032
ワークステーション・セクション１	2034
ワークステーション・ログ１	2036
・・・	2038
ワークステーション・ログ１	2040
ワークステーション状況ファイル	2042
・・・	2044
ワークステーション・セクション１	2046

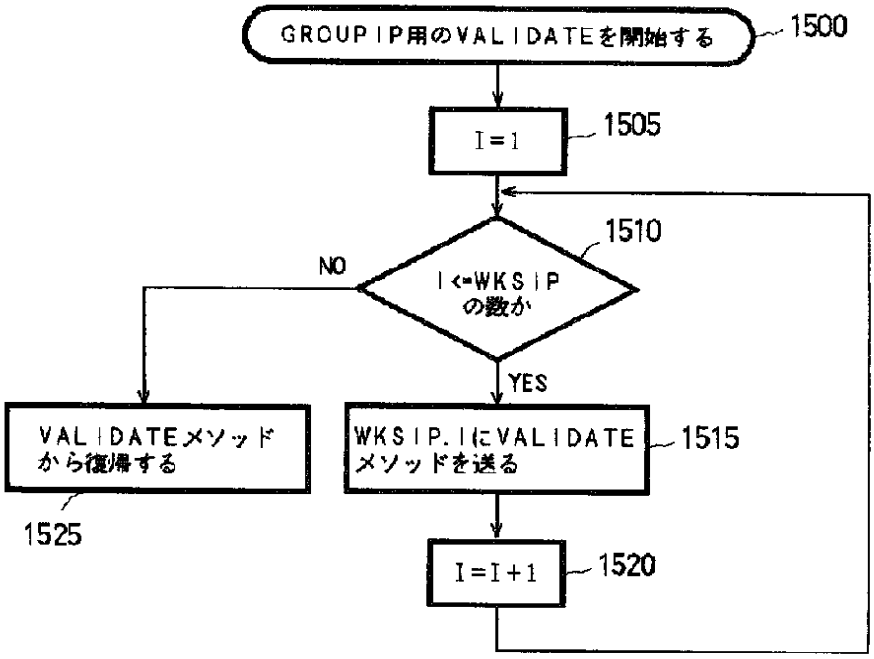
【図１４】



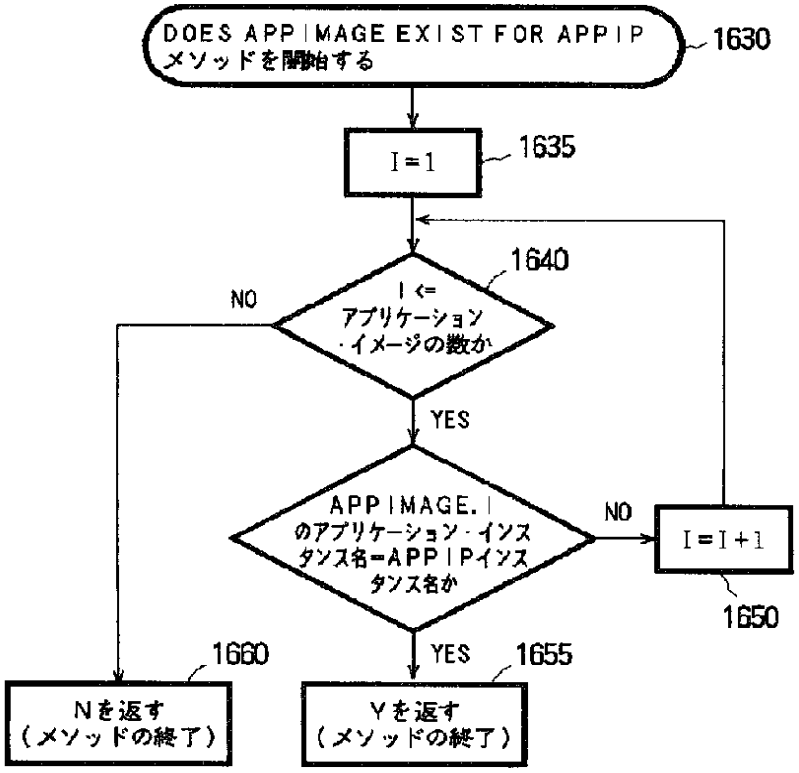
【図２４】

ＰＲＥコマンド・ファイル	2070
接続導入１	2072
・・・	2074
接続導入Ｎ	2076
ネットワーク導入使用可能性	2078

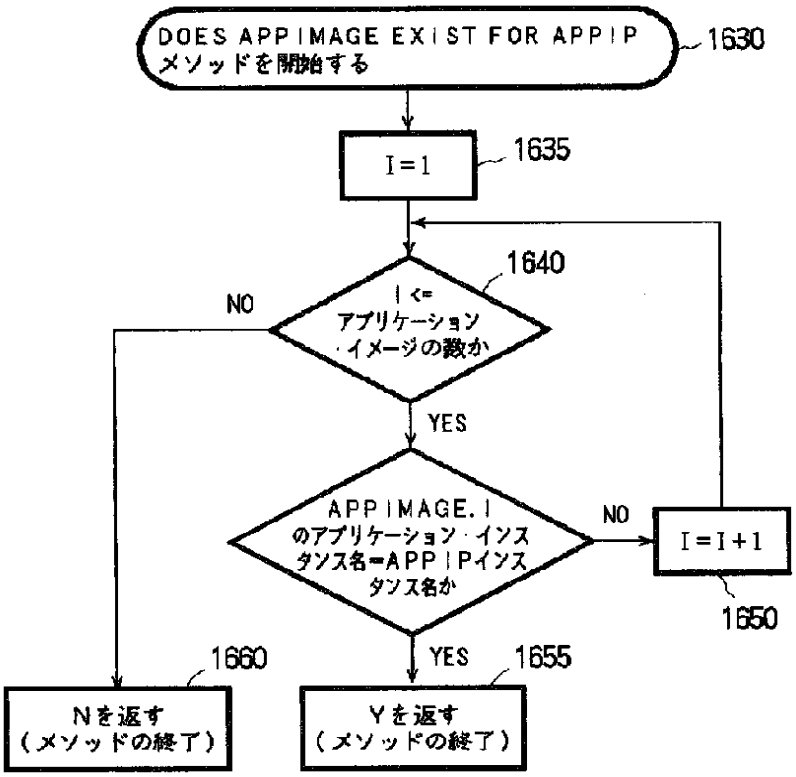
【図１８】


GROUP IP用のVALIDATEを開始する
1500
I=1
1505
1510
NO
I<=WKSIP
の数か
YES
VALIDATEメソッド
から復帰する
1525
WKSIP.IにVALIDATE
メソッドを送る
1515
I=I+1
1520

【図１９】


DOES APPIMAGE EXIST FOR APPIP
メソッドを開始する
1630
I=1
1635
1640
NO
I<=
アプリケーション
・イメージの数か
YES
APPIMAGE.I
のアプリケーション・インス
タンス名=APPIPインス
タンス名か
NO
I=I+1
1650
YES
1655
1660
Nを返す
（メソッドの終了）
Yを返す
（メソッドの終了）

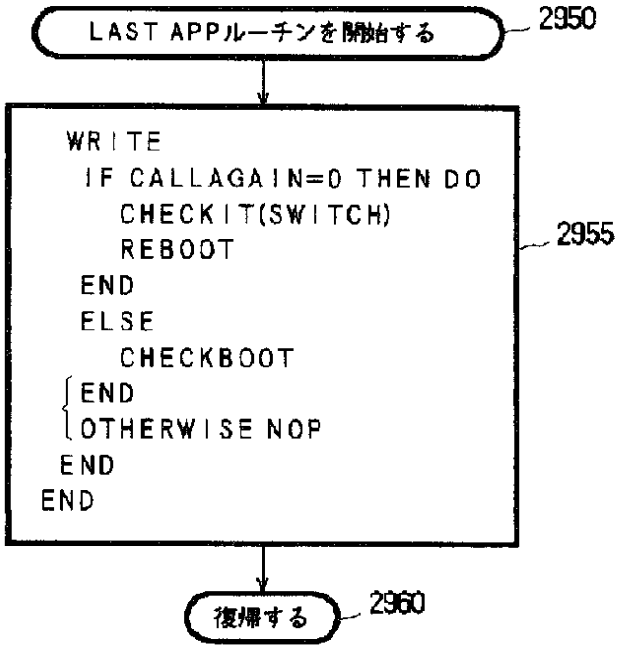
【図２０】



【図２１】

ＰＯＦファイル	2000
計画セクション	2002
シーケンス	2004
ＡＰＰセクション１	2006
・・・	2008
ＡＰＰセクションＮ	2010
ワークステーション・セクション１	2012
・・・	2014
ワークステーション・セクションＮ	2016
サーバ・セクション１	2018
・・・	2020
サーバ・セクションＮ	2022

【図３５】



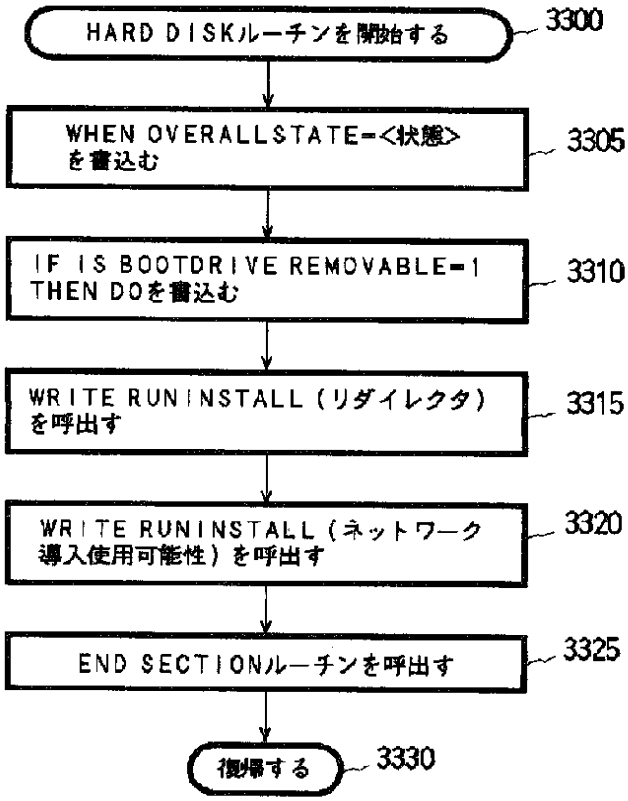
【図２５】

ＰＯＳＴコマンド・ファイル	2080
接続除去１	2082
・・・	2084
接続除去１	2086
ネットワーク導入使用可能除去	2088

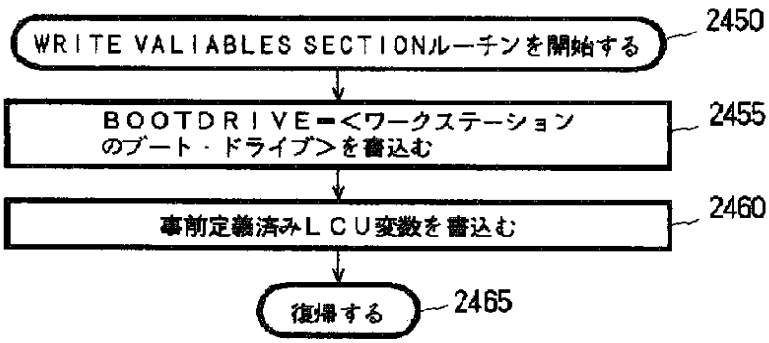
【図２６】

ＬＣＵコマンド・ファイル	2090
セットアップ・セクション	2092
接続セクション	2094
変数セクション	2096
製品データ・セクション	2098
導入セクション	2100
ＣｈｅｃｋＩＴ手続き	2102
基本ＬＣＵ手続きセクション	2104

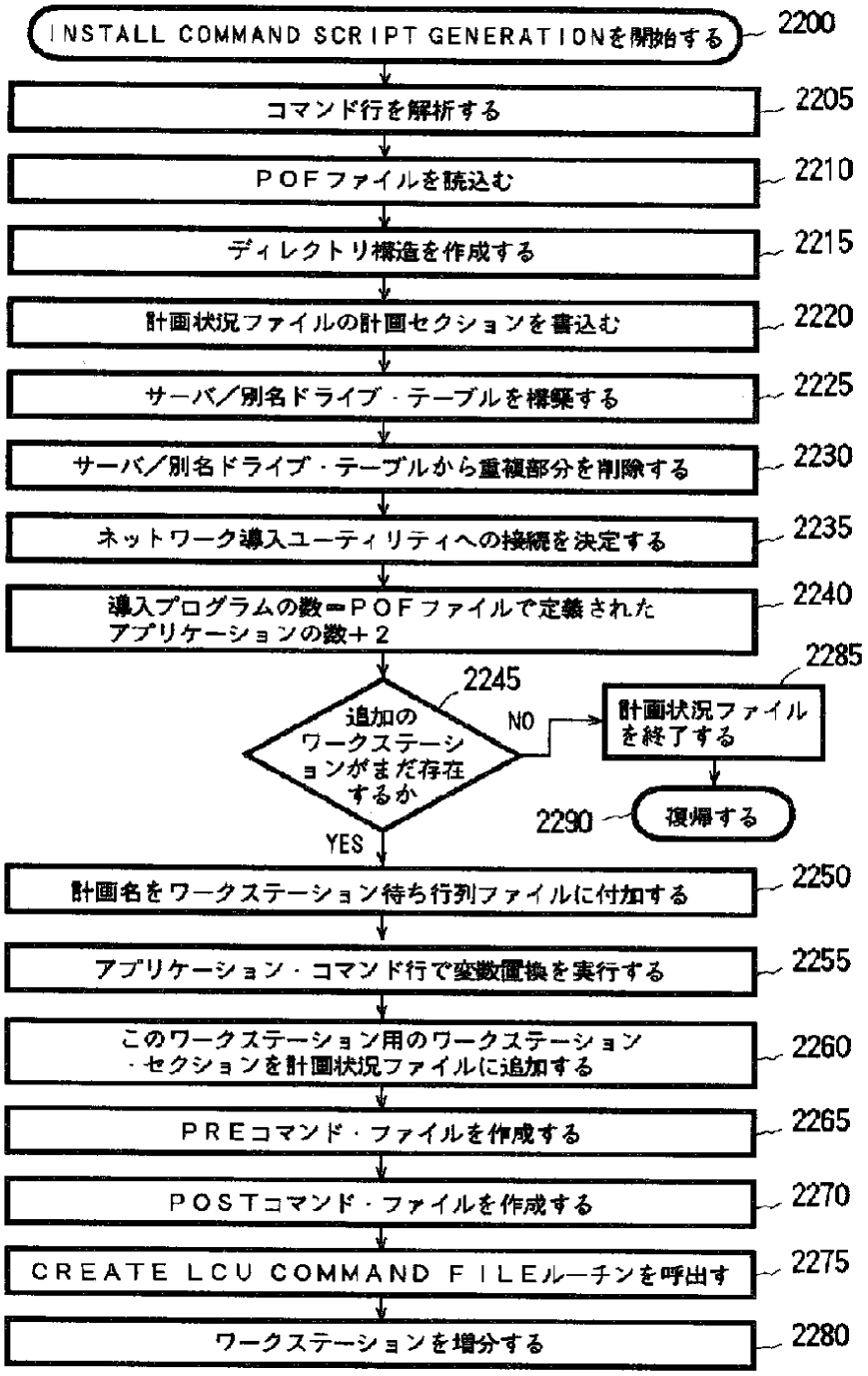
【図３９】



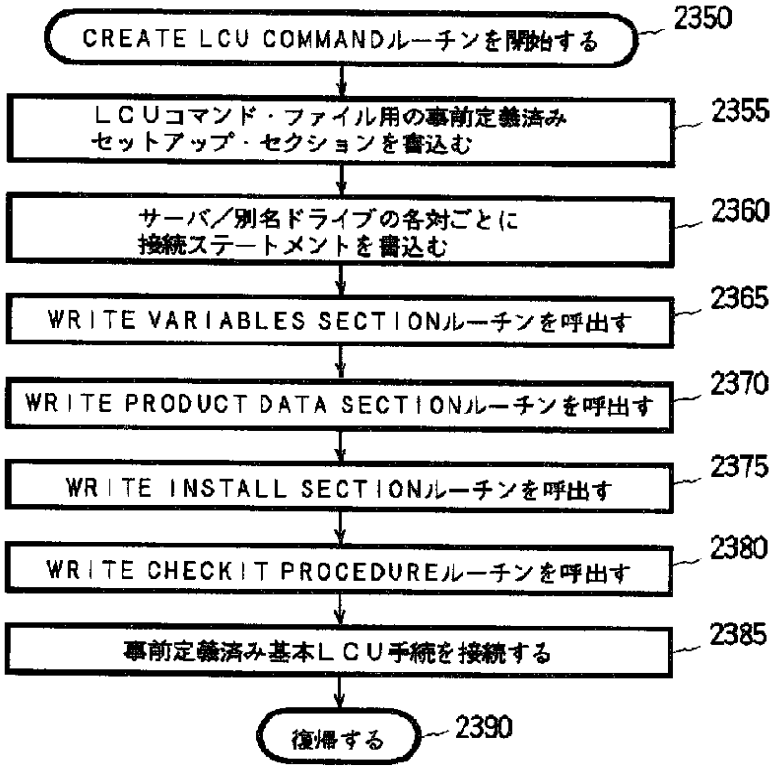
【図２９】



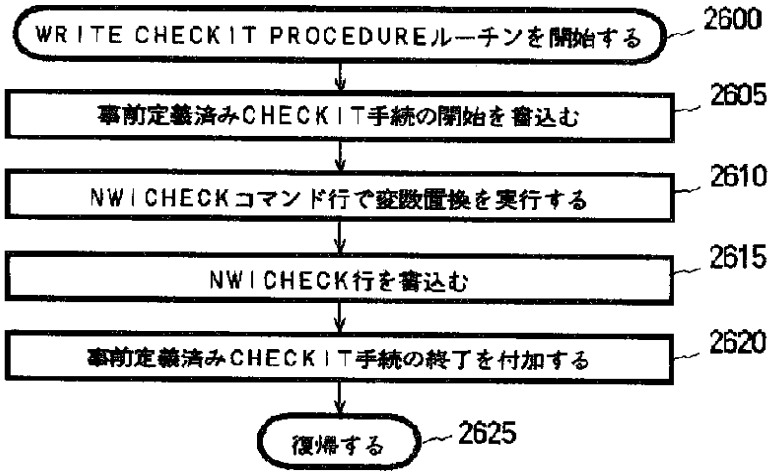
【図２７】


INSTALL COMMAND SCRIPT GENERATIONを開始する
2200
コマンド行を解析する
2205
ＰＯＦファイルを読込む
2210
ディレクトリ構造を作成する
2215
計画状況ファイルの計画セクションを書込む
2220
サーバ／別名ドライブ・テーブルを構築する
2225
サーバ／別名ドライブ・テーブルから重複部分を削除する
2230
ネットワーク導入ユーティリティへの接続を決定する
2235
導入プログラムの数＝ＰＯＦファイルで定義された
アプリケーションの数＋２
2240
2245
追加の
ワークステーシ
ョンがまだ存在
するか
NO
YES
2285
計画状況ファイル
を終了する
2290
復帰する
計画名をワークステーション待ち行列ファイルに付加する
2250
アプリケーション・コマンド行で変数置換を実行する
2255
このワークステーション用のワークステーション
・セクションを計画状況ファイルに追加する
2260
ＰＲＥコマンド・ファイルを作成する
2265
ＰＯＳＴコマンド・ファイルを作成する
2270
ＣＲＥＡＴＥ ＬＣＵ ＣＯＭＭＡＮＤ ＦＩＬＥルーチンを呼出す
2275
ワークステーションを増分する
2280

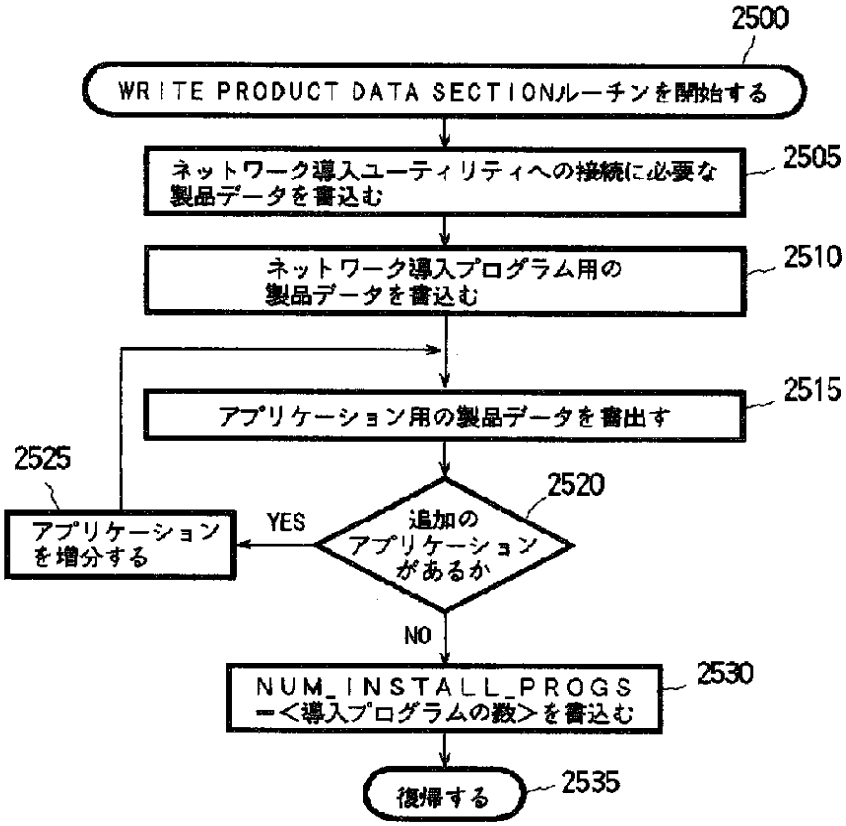
【図２８】


CREATE LCU COMMANDルーチンを開始する
2350
ＬＣＵコマンド・ファイル用の事前定義済み
セットアップ・セクションを書込む
2355
サーバ／別名ドライブの各対ごとに
接続ステートメントを書込む
2360
WRITE VARIABLES SECTIONルーチンを呼出す
2365
WRITE PRODUCT DATA SECTIONルーチンを呼出す
2370
WRITE INSTALL SECTIONルーチンを呼出す
2375
WRITE CHECKIT PROCEDUREルーチンを呼出す
2380
事前定義済み基本ＬＣＵ手続を接続する
2385
復帰する
2390

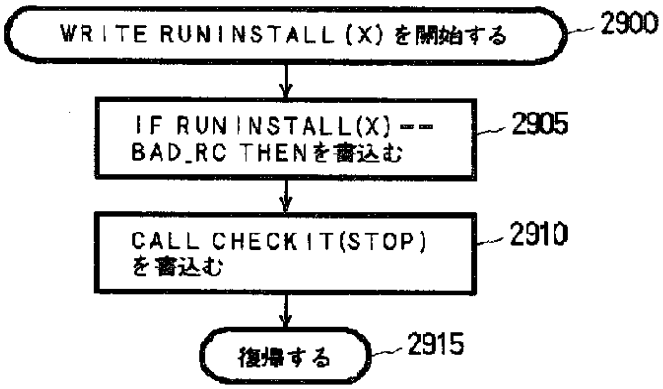
【図３１】


WRITE CHECKIT PROCEDUREルーチンを開始する
2600
事前定義済みCHECKIT手続の開始を書込む
2605
NWICHECKコマンド行で変数置換を実行する
2610
NWICHECK行を書込む
2615
事前定義済みCHECKIT手続の終了を付加する
2620
復帰する
2625

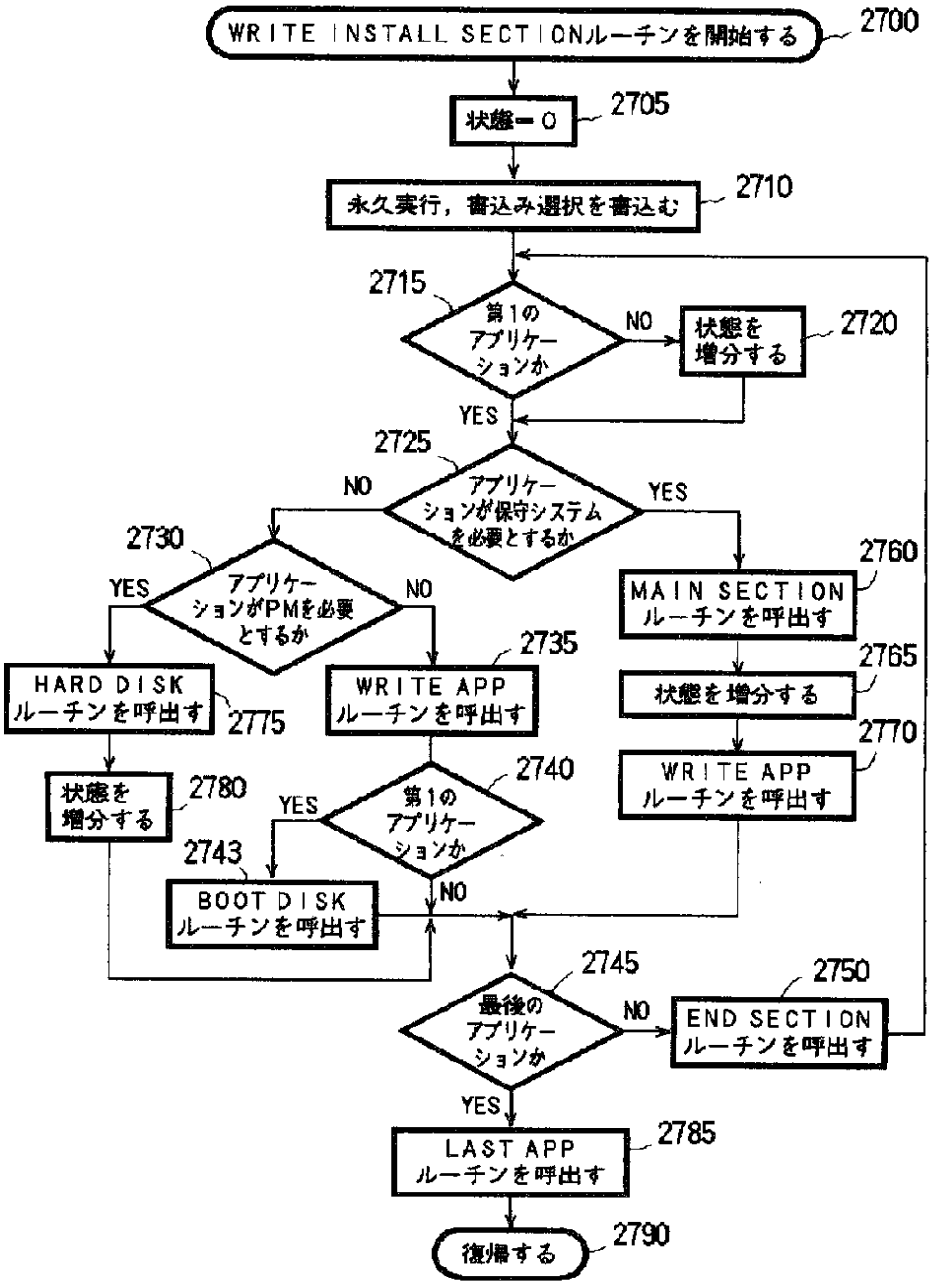
【図３０】


2500
WRITE PRODUCT DATA SECTIONルーチンを開始する
ネットワーク導入ユーティリティへの接続に必要な製品データを書込む
2505
ネットワーク導入プログラム用の製品データを書込む
2510
アプリケーション用の製品データを書出す
2515
2525
アプリケーションを増分する
YES
追加のアプリケーションがあるか
2520
NO
NUM_INSTALL_PROGS =<導入プログラムの数>を書込む
2530
復帰する
2535

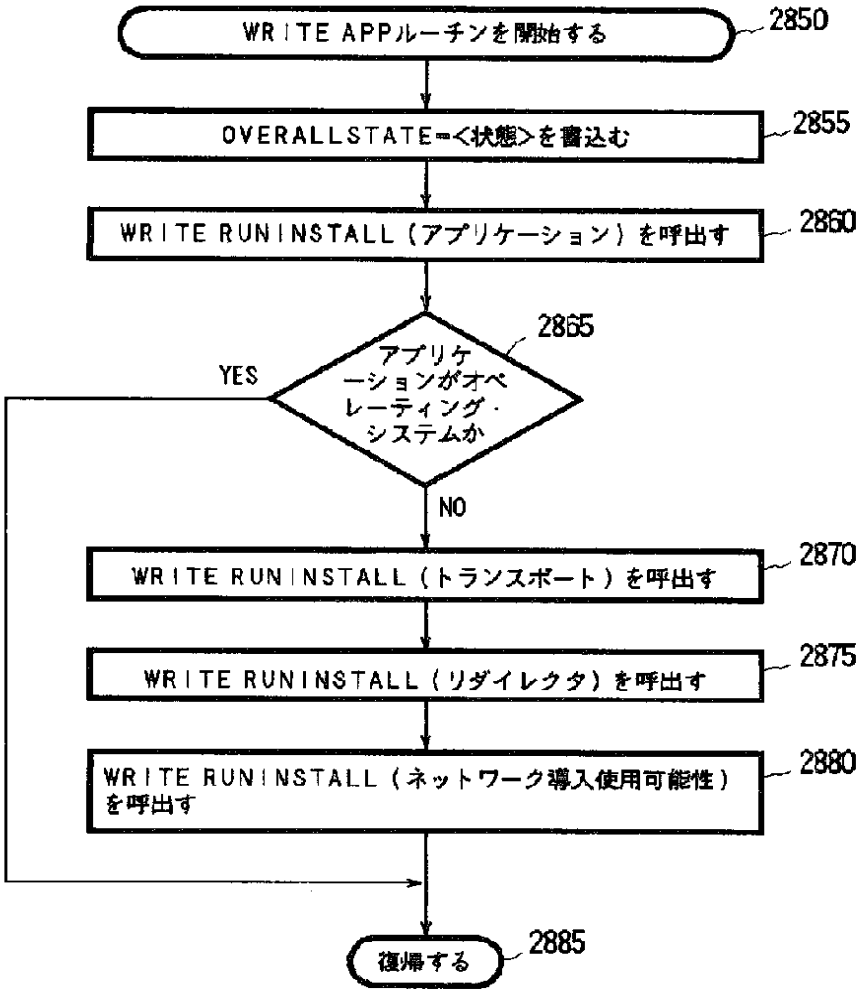
【図３４】


WRITE RUNINSTALL (X) を開始する
2900
IF RUNINSTALL(X) == BAD_RC THENを書込む
2905
CALL CHECKIT(STOP)を書込む
2910
復帰する
2915

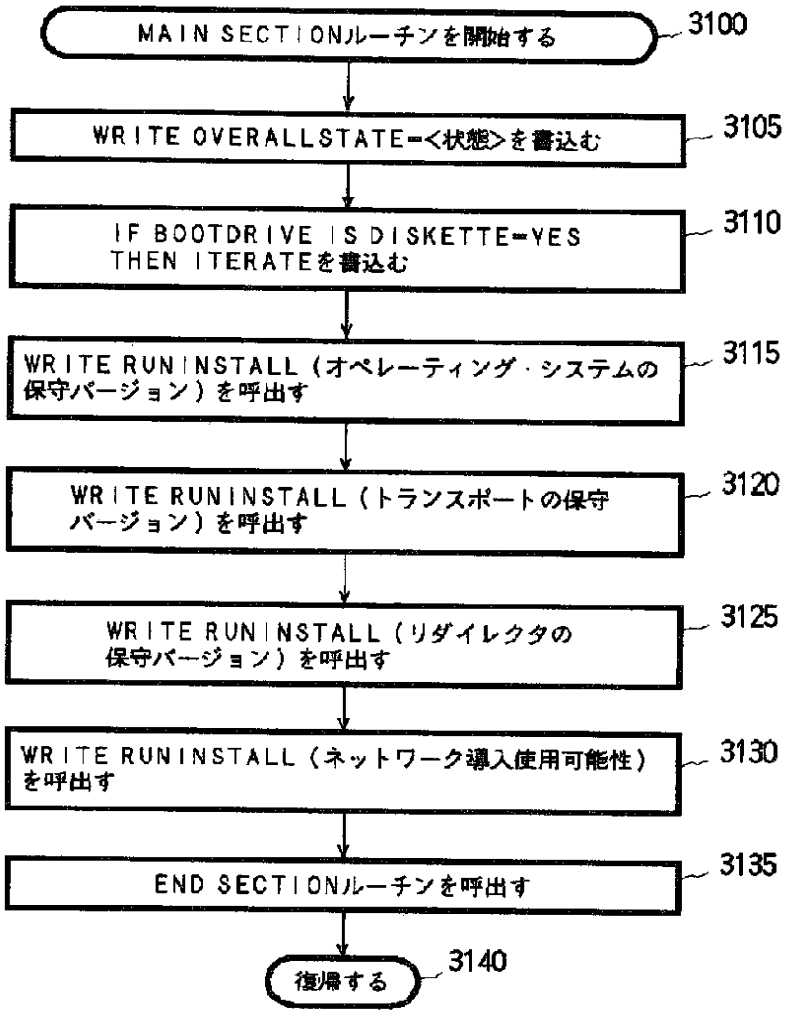
【図３２】


WRITE INSTALL SECTIONルーチンを開始する
2700
状態＝０
2705
永久実行，書込み選択を書込む
2710
2715
第1のアプリケーションか
NO
状態を増分する
2720
YES
2725
アプリケーションが保守システムを必要とするか
NO
YES
2730
アプリケーションがPMを必要とするか
YES
NO
2760
MAIN SECTIONルーチンを呼出す
2765
状態を増分する
2770
WRITE APPルーチンを呼出す
HARD DISKルーチンを呼出す
2775
WRITE APPルーチンを呼出す
2735
2740
第1のアプリケーションか
YES
NO
2743
BOOT DISKルーチンを呼出す
状態を増分する
2780
2745
最後のアプリケーションか
NO
2750
END SECTIONルーチンを呼出す
YES
LAST APPルーチンを呼出す
2785
復帰する
2790

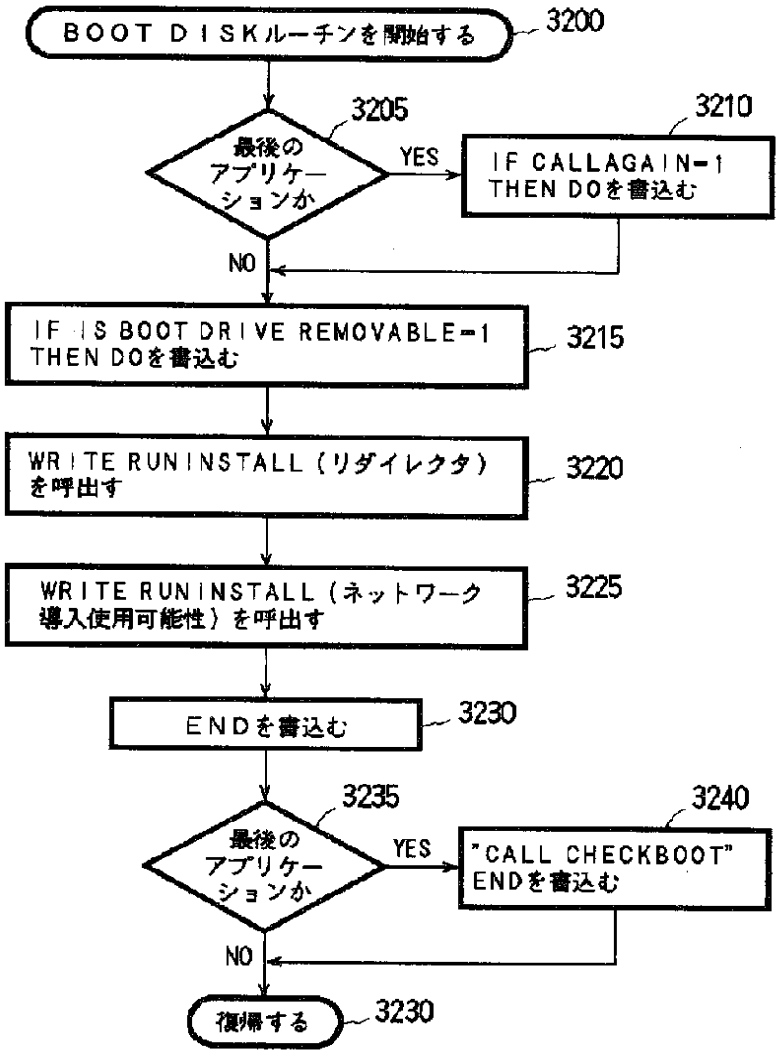
【図３３】


WRITE APPルーチンを開始する
2850
OVERALLSTATE=<状態>を書込む
2855
WRITE RUNINSTALL（アプリケーション）を呼出す
2860
アプリケーションがオペレーティング・システムか
2865
YES
NO
WRITE RUNINSTALL（トランスポート）を呼出す
2870
WRITE RUNINSTALL（リダイレクタ）を呼出す
2875
WRITE RUNINSTALL（ネットワーク導入使用可能性）を呼出す
2880
復帰する
2885

【図３７】


MAIN SECTIONルーチンを開始する
3100
WRITE OVERALLSTATE=<状態>を書込む
3105
IF BOOTDRIVE IS DISKETTE=YES
THEN ITERATEを書込む
3110
WRITE RUNINSTALL（オペレーティング・システムの保守バージョン）を呼出す
3115
WRITE RUNINSTALL（トランスポートの保守バージョン）を呼出す
3120
WRITE RUNINSTALL（リダイレクタの保守バージョン）を呼出す
3125
WRITE RUNINSTALL（ネットワーク導入使用可能性）を呼出す
3130
END SECTIONルーチンを呼出す
3135
復帰する
3140

【図３８】



フロントページの続き

(72)発明者　バーバラ・ジャン・ジェンセン
アメリカ合衆国78727－6075　テキサス州　オースチン　クレーグ・ドライブ　4902

(72)発明者　セオドア・ジャック・ロンドン・シュレーダー
アメリカ合衆国78613　テキサス州シーダー・パーク　シェイディ・ブルック・レーン　1704